Markov Chain Methods in Protein Folding Dynamics

Dissertation
zur
Erlangung der naturwissenschaftlichen Doktorwürde
(Dr. sc. nat.)

vorgelegt der
Mathematisch-naturwissenschaftlichen Fakultät
der
Universität Zürich

von
Riccardo Scalco
aus
Italy

Promotionskomitee

Prof. Dr. Amedeo Caflisch
Prof. Dr. Ben Schuler

Zürich 2012

Inhaltsverzeichnis

Abstract

Markov State Models (MSM) are a useful tool to describe the kinetics of a folding process: the molecular dynamics simulation is linked to a complex network, whose nodes and links are clusters of snapshots and transitions between them, respectively. The great advantage of a complex network description is to consider, for the quantification of kinetics observables, dynamic pathways that do not appear on the MD simulation, but that are of high probability.

The following thesis presents two problems that emerge in a Markov Chain description of a MD simulation, namely, the introduction of a statistical bias with the combining of multiple independent trajectories (chapter 2), and the extraction of free-energy barriers between states from MD data (chapter 3). Both of the problems are addressed in a MSM framework, with a particular emphasis on cut-based free-energy profiles, a method developed in the last years.

During my works on Markov Chain methods on MD data I had the occasion to develop a software able to create, manipulate and analyze Markov chains, it makes easier to calculate many observables, like the steady state, the mean first passage times, the mean exit times, and so on. The software, called **Pykov**, is freely available on the web, and its documentation is presented as appendix at the end of the thesis.

Chapter 4 describes the work I did on my first year. It is a work-in-progress project where it is investigated the behaviour of a simplified 56-residue sequence, with a restricted alphabet of only three amino acids. The guideline of the project is to observe significant changes on the protein thermodynamics and kinetics with the change of the alphabet size.

Finally, some future developments are outlined on the conclusions.

Zusammenfassung

Markov State Modelle sind ein äusserst nützliches Werkzeug um biologische Prozessse wie zum Beispiel Proteinfaltung zu untersuchen. Die Proteindynamik wird mittels eines komplexen Netzwerks beschrieben, dessen Knoten die Protein Konformationen darstellen und dessen Kanten die Transitionen zwischen den einzelnen Konformationen repräsentieren. Jede signifikante Frage, welche in der Biochemie formuliert wird, sollte in den Markov'schen Formalismus übersetzt werden. Diese Thesis zeigt, wie man biochemische Probleme im Feld der Proteine und deren dynamischem Verhalten, auf Basis der Markov State Modelle formulieren und lösen kann. Insbesondere wird die statistische Bias diskutiert, welche entsteht, wenn multiple unabhängige Molekulardynamik Trajektorien kombiniert werden. Ebenfalls wird speziell auf die genaue Extraktion von freien Energie Barrieren zwischen verschiedenen Protein Konformationen eingegangen.

Chapter 1

Introduction

1.1 MD simulations

The fundamental problem of the *dynamics* is to solve the equation of motion, i.e. to link the implicit description of the motion laws to an explicit solution that describes the trajectory of the physical system in its phase space. In the presence of non-linear dynamical systems, or of linear systems with a too high number of degrees of freedom, there are not analytic solutions and the help of computer simulations is invoked, this is the case of protein folding dynamics.

Molecular dynamics simulations are prepared thinking at mainly two ingredients, which could vary depending on the system under investigation: the equation of motion and the algorithm of integration. The output, i.e. the MD simulation, is a time ordered list of snapshots, namely configurations of the system (or points in its phase space), because of this the *ergodic hypothesis* plays in molecular dynamics of protein structures a fundamental role. The well-know ergodic theorem of Birkhoff states that the limit $\lim_{C\to\infty}\int_0^C \frac{1}{C} f(P_t)dt$, which is the mean of the observable f along a trajectory P_t, exists almost everywhere (i.e. with probability 1) and does not depend on the initial point P_0. If the system is ergodic, then with probability 1 the above limit is equal to the average of the function f in the whole phase space: $\frac{1}{V}\int_V f(P)dV$. The theorem is valid in the case of the metric indecomposability of the phase space V, namely in the case that the trajectories pass through almost all point of V. Ergodic hypothesis is fundamental in molecular dynamics, because otherwise every observable calculated along a trajectory would depend strongly on the initial conditions.

In order to describe the MD simulation by means of a Markov state model (MSM), it is often useful to group together the nodes in disjoint subsets (called clusters or nodes). Such clustering procedure could be done in many different ways, which one is better depends on the observables under study, and it has the main result to translate the MD simulation in a symbolic trajectory, i.e. the temporal list of the cluster names. Once the symbolic trajectory is defined, it could be related to a complex network and, considering the statistics of the transitions between the nodes, to a Markov model. The great advantage of a markovian description is to consider, for the quantification of kinetics observables, transitions that do not appear on the MD simulation, but that are of high probability.

1.2 Markov state models

Definitions and theorems concerning Markov chains are presented in this paragraph. In particular it is shown the procedure used to define a Markov model from a symbolic trajectory. Only basic theory is presented, a complete description is provided in [1].

1.2.1 Definitions

Definition 1.2.1 (*trajectory* t)**.** Let $\mathcal{K}$ be a finite alphabet of characters (integers in $[1, N]$ in our case). A trajectory $t = (t_1, t_2, t_3, \ldots, t_L)$, where $t_k \in \mathcal{K}$, may be thought as a function on the

one dimensional array $p = (1, 2, \ldots, L)$ of time steps. The symbols of the alphabet are often called *nodes* or *clusters*.

Example 1.2.1. Let $\mathcal{K} = \{1, 2, 3\}$, a possible trajectory is

$$(1, 1, 1, 2, 1, 2, 3, 3, 2, 1, 3, 2)$$

Definition 1.2.2 (*links set* $L(t)$)**.** Give a trajectory t, of length L, the set of its links ($L(t)$) is the set of all the one step transitions along the trajectory. Note that the number of such links is equal to $L - 1$.

Example 1.2.2. Let $t = (1, 1, 1, 2, 1, 3)$, so the set of its links is

$$L(t) = \{(1, 1), (1, 1), (1, 2), (2, 1), (1, 3)\}$$

Definition 1.2.3 (*transitions rate probability:* q_{ij})**.** Given a trajectory t, q_{ij} is the probability to find a link (i, j) in $L(t)$

Example 1.2.3. Taken $L(t) = \{(1, 1), (1, 1), (1, 2), (2, 1), (1, 3)\}$, then

$$q_{1,1} = \frac{2}{5}$$

Definition 1.2.4 (*probability over the nodes:* p_i)**.**

$$p_i = \sum_j q_{ij}$$

Observation 1.2.1.

$$\sum_i p_i = \sum_{i,j} q_{ij} = 1$$

Definition 1.2.5 (*transition probabilities:* p_{ij})**.**

$$p_{ij} = \frac{q_{ij}}{p_i}$$

Observation 1.2.2. The transition probabilities p_{ij} form a stochastic matrix, i.e. $p_{ij} \geqslant 0$ and

$$\sum_j p_{ij} = \sum_j \frac{q_{ij}}{p_i} = \frac{\sum_j q_{ij}}{p_i} = 1$$

The matrix which entries are p_{ij} is called *transition matrix*, see figure 1.1.

$$P = \begin{pmatrix} p_{1,1} & 0 & p_{1,3} \\ p_{2,1} & 0 & p_{2,3} \\ p_{3,1} & p_{3,2} & 0 \end{pmatrix}$$

$$\sum_j p_{i,j} = 1 \quad \forall i$$

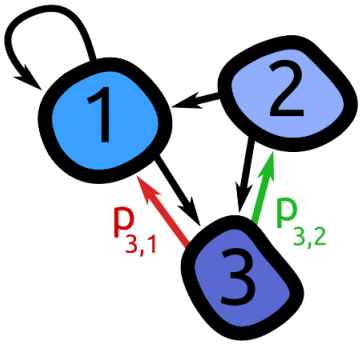


Figure 1.1: A transition matrix with the corresponding graph.

1.2.2 A bit of theory

A *finite Markov chain* is characterized by a transition matrix T and a initial distribution $z(0) = (z_1, z_2, \ldots, z_n)$. The probability distribution after m steps is

$$z(m) = z(0)T^m$$

Definition 1.2.6 (*stationary distribution*)**.** A *stationary distribution*, for the Markov chain with transition matrix T, is a probability distribution π such that:

$$\pi_i \geqslant 0$$
$$\sum_i \pi_i = 1$$
$$\pi T = \pi$$

Theorem 1.2.1. Every finite Markov chain has a (at least one) stationary distribution.

Definition 1.2.7 (*transition digraph*)**.** Let T a stochastic $n \times n$ matrix. We associate to T a directed graph (digraph) $G = (V, E)$, where

$$V = \{1, 2, \ldots, n\} \qquad E = \{(i, j) : t_{ij} \neq 0, i, j \in V\}$$

If T is the transition matrix of the finite Markov chain $\mathcal{M}$, the *nodes* of G represent the states of $\mathcal{M}$ and the *edges* the feasible transitions weighted by p_{ij}.

Definition 1.2.8 (*path* and *cycle* in graphs)**.** A *path* in a graph is a sequence of nodes and edges such that from each of its nodes there is an edge to the next vertex in the sequence. A *cycle* is a path such that the start vertex and end vertex are the same.

Definition 1.2.9 (*strongly connected digraph*)**.** A directed graph is *strongly connected* if, given any two nodes, there exists a path from the first vertex to the second. The associated Markov chain is called *irreducible*.

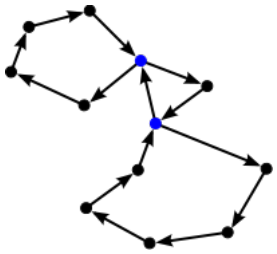


Figure 1.2: Example of a cycle

Theorem 1.2.2. A irreducible Markov chain has exactly one stationary distribution π, $\pi T = \pi$, which entries are all positive, $\pi_i > 0 \forall i$.

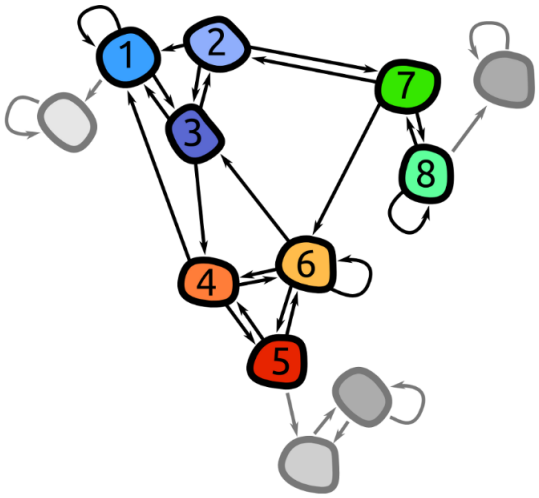


Figure 1.3: A network with four strongly connected components. Usually, the biggest component contains most of the data.

Definition 1.2.10 (*aperiodic digraph*)**.** Given a digraph G, the greatest common divisor of the lengths of every cycle in G is called the period of G. G is *aperiodic* if its period is one.

Definition 1.2.11 (*ergodic Markov chain*)**.** A irreducible aperiodic Markov chain is called *ergodic*.

Theorem 1.2.3. If T is the transition matrix of an ergodic Markov chain then the powers of T converge:

$$\lim_{m \to \infty} T^m = K$$

and the rows of K are equal. Let z an arbitrary initial distribution, and π the stationary distribution, then

$$\lim_{m \to \infty} zT^m = \pi$$

In other words, if a finite Markov chain is ergodic then from any initial distribution the process will approach the unique stationary distribution.

1.3 Equilibrium distribution from distributed computing

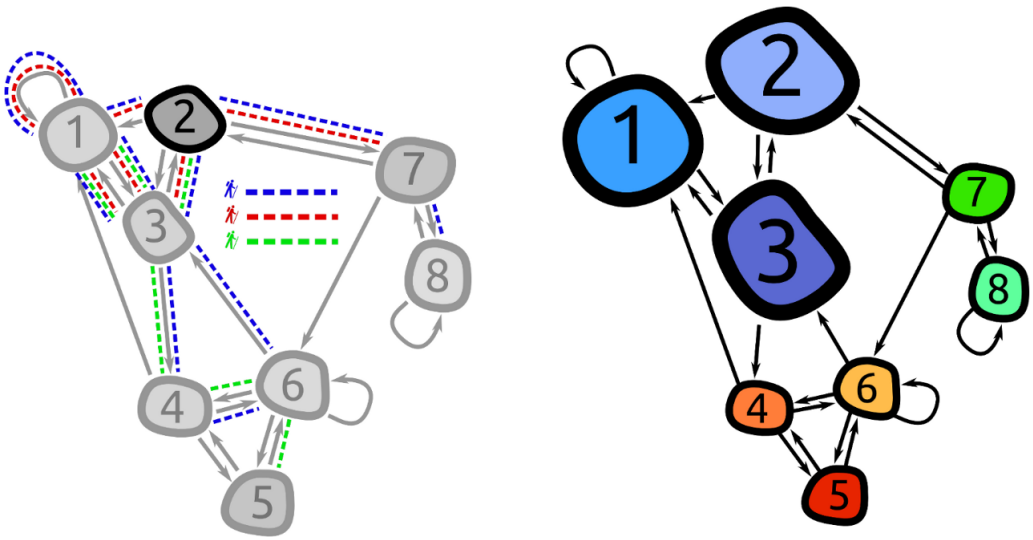


Figure 1.4: A multiple trajectory analysis on a graph. All the trajectories start from state 2, with the consequence that the probability distribution is biased.

This paragraph introduces briefly the argument presented in chapter 2.

Research in computational biochemistry benefits from the progress on computing power, in particular from the possibility to design distributed computing simulations of protein folding. When multiple short MD simulations (the simulations length is comparable to their number) are carried out from a single structure or from a set of conformations, a significant statistical bias is usually present on the data. The bias arises from the fact that the initial conformations are not Boltzmann distributed. On the MSM generated from biased MD data the probability distribution of the nodes is also biased (see figure 1.4).

In relation to the statistical bias, the problems addressed in chapter 2 are the following:

- How far is the biased distribution from the correct one?
- How does it depend on the initial conformations and on the number and length of the simulations?
- How to get the correct distribution from the biased one?

The answers at the above problems are based on the following assumption: the transition probabilities extracted from the biased data are correct, we are confident in such assumption because transition probabilities p_{ij} are conditional probabilities: $p_{ij} \equiv q_{ij}/p_i$. The first two questions are answered resorting to the fundamental matrix Z of the Markov Chain, in particular one can easily see that the bias is much stronger when the trajectories length is shorter and the number of simulation is bigger, as expected. Moreover it is not possible to solve the bias imposing detailed balance by means of symmetrization of the transition rate probabilities q_{ij}. The answer at the last question is straightforward, it is enough to calculate the steady state

distribution of the chain. In relation to this last point, in order to calculate the steady state the chain must be ergodic and often, because of technical problems, it is not. In chapter 2 it is also presented a possible way to get an ergodic chain from a non-ergodic one: it is suggested to remove the minimum number of links in order to have an ergodic network. This task is easily solved by the Tarjan algorithm, which is able to subdivide the whole graph in its strongly connected components (see figure 1.3).

1.4 Ultrametricity and Free-Energy barriers (cFEP methods)

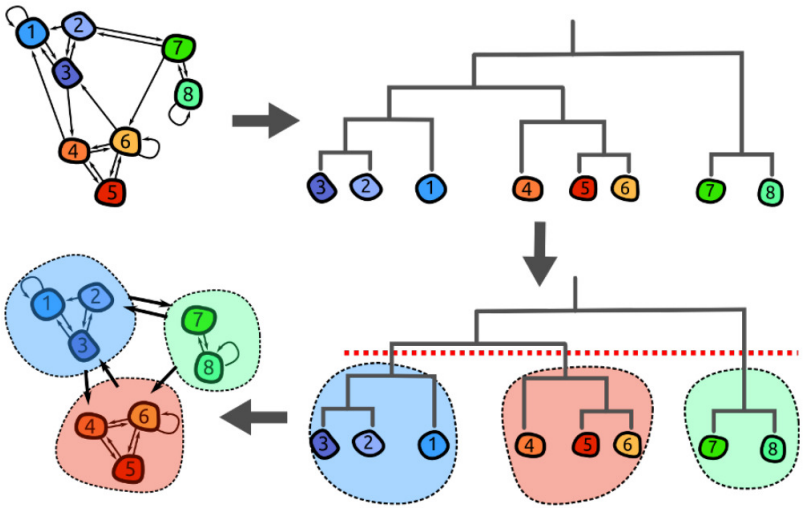


Figure 1.5: The ultrametricity of the cut-based free-energy generates a hierarchical tree from the complex network, with the consequence that a clustering of nodes in states is unambiguously built. The cbFE describes the free-energy landscape, then the clustering is based on the system kinetics.

This paragraph introduces briefly the argument presented in chapter 3.

In order to compare kinetic observables (for example rate constants), calculated within a Markov state model of a MD system, with the corresponding values calculated in transition state theory (TST), Kramers' theory, or with experimental observables, concepts like *state*, *transition state free-energy* and *free-energy barrier* have to be defined on a Markov framework.

Chapter 3 presents how the definition of *cut-based free-energy* (cbFE) is able alone to create the semantic bridge between Markov theory and TST or Kramers' theory. In particular, it is shown that the cbFE defines an ultrametric distance on the set of nodes, which permits to i) clusterize the network in basins (or states), ii) calculate the transition state free energy between two nodes, iii) estimate the activation free energy to exit from a basin and iv) calculate the rate constant between two basins.

Chapter 3 is devoted to explain the four points above. The novelties are mainly two: the recognition of the cbFE as ultrametric distance and the extension of the cut-based free-energy profile (cbFEP) method. The former is useful to subdivide the network in basins (figure 1.5), the

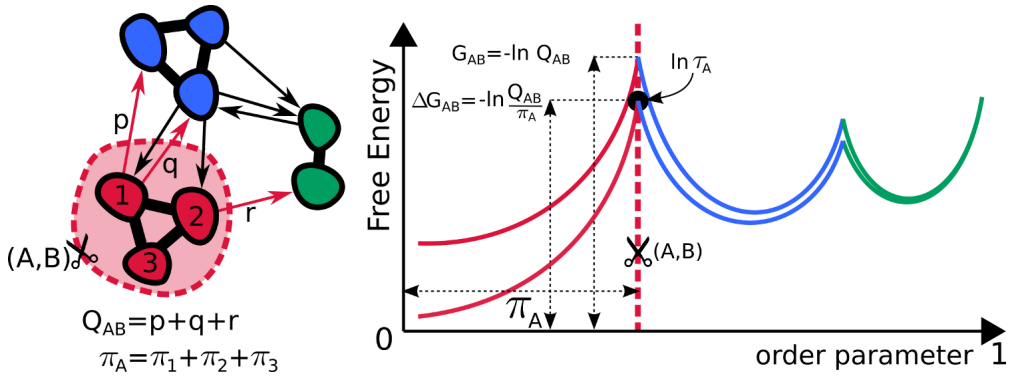


Figure 1.6: (Left) Schematic representation of a MSM consisting of eight nodes grouped in three states (basins). Basin A, of probability π_A, is selected by means of a cut. (Right) Graphical representation of the cbFEP related to the MSM on the left. The cut (A, B) separates the profiles at the first peak, where the values $G_{A,B}$, $\Delta G_{A,B}$ and τ_A indicate the free energy of the transition state, the activation free energy, and the mean time to exit from state A, respectively. The superposition of τ_A and $\Delta G_{A,B}$ indicates that the equilibration in A is much faster than the mean time to escape.

latter is useful to check for the existence of a basin for which the equilibration is much faster than the mean time to escape from it, an assumption of both TST and Kramers' theory (figure 1.6).

1.4.1 The diffusivity test

MD systems are assumed to have a diffusive dynamics, the cbFEP is a tool useful also to check if such assumption holds on the system under investigation. In the following paragraph it is briefly presented the theory behind the diffusivity test.

Let Ω the system phase space, the element $x(t) \in \Omega$ describes the state of the system at time t. In molecular dynamics Ω is usually identified with $\mathbb{R}^{6N}$ [2], with N the number of atoms. We assume the dynamics is a homogeneous Markov process [3], namely the probability $p(s, x, t, y)$ to move from state $x(s)$ to state $y(t)$ satisfies at the Kolmogorov-Chapman equation:

$$p(s, x, t, y) = \int_\Omega p(s, x, u, z)p(u, z, t, y)dz \qquad s < u < t$$

We assume also the dynamics describes a diffusion process and the existence of a quench interval τ, namely the time step between two consecutive observations, such that the drift coefficient equals to zero and diffusion coefficient is constant (named σ^2). In formulas, the latter assumption

is

$$\int_{|y-x|\leq\epsilon}(y-x)p(s,x,s+\tau,y)dy=0$$
$$\int_{|y-x|\leq\epsilon}(y-x)^2p(s,x,s+\tau,y)dy=\sigma^2\tau+o(\tau)$$

for ϵ small enough. Under the quench interval τ, the process appears as a Brownian motion. The equation defining a Brownian motion is the well-known diffusion equation [3]

$$\partial_t p(s,x,s+t,y)=\frac{\sigma^2}{2}\nabla^2 p(s,x,s+t,y)$$

with the fundamental solution

$$p(s,x,s+t,y)=(2\pi\sigma^2 t)^{-1/2}e^{-(y-x)^2/(2\sigma^2 t)}$$

Let (A,B) a bipartition of Ω in two connected subspaces. The cut free-energy profile (cFEP) defines the free-energy of the transition state between A and B [4]. In units of k_BT, it is defined as

$$F_{AB}(\tau)=-\log P_{AB}(\tau)$$

where $P_{AB}(\tau)$ is the probability to observe the system in $x(t)\in A$ and $x(t+\tau)\in B$. Let $\phi(x,t)$ the states probability distribution at time t, the probability P_{AB} is calculated with

$$P_{AB}(\tau)=\int_{x\in A}\phi(x,t)\int_{y\in B}p(t,x,t+\tau,y)dydx$$

Under the assumption that $\phi(x,t)$ is constant in a interval of order $\sqrt{\sigma^2\tau}$, corresponding to the range in which $p(t,x,t+\tau,y)$ is different from zero, the dependence of P_{AB} and F_{AB} from the quench interval τ satisfies at the relations [5]

$$P_{AB}(2\tau)\simeq\sqrt{2}P_{AB}(\tau)$$
$$F_{AB}(2\tau)\simeq F_{AB}(\tau)-\frac{\log 2}{2}$$

In this work, we derive from the continuous Markov process a Markov model discrete in space and time. Such discretization procedure introduce a systematic error due to the partitioning of the phase space [6], it derives from the original Markov process a lumped process with could be non-markovian [1]. Only an accurate choice for the partitioning scheme and for the lag-time permits to shift from the continuous description to a discrete description [6]. Once the discrete Markov model, with transition matrix T, is obtained, we use the above result $F_{AB}(2\tau)\simeq F_{AB}(\tau)-\log 2/2$ to find the optimal quench interval at which the system displays a diffusive regime. Namely, we calculated T^τ for $\tau=1,2,4,8,\ldots$, at the optimal τ we expect $F_{AB}(2\tau)\simeq F_{AB}(\tau)-\log 2/2$, then the Markov model used to describe the system kinetics is T^τ.

Supplementary informations. Here we derive the result $P_{AB}(2\tau) \simeq \sqrt{2}P_{AB}(\tau)$ in the special case of a one-dimensional system, $\Omega = \mathbb{R}$. Let $A = \{x : x < k\}$ and $B = \{x : x > k\}$, then $P_{AB}(\tau)$ is

$$\begin{aligned} P_{AB}(\tau) \equiv P_k(\tau) &= \int_{x\in A} \phi(x,t) \int_{y\in B} p(t,x,t+\tau,y)dydx = \\ &= \int_{-\infty}^{k} \phi(x,t) \int_{k}^{\infty} p(t,x,t+\tau,y)dydx \end{aligned}$$

We introduce the new variable $w = y - x$. From $x \in (-\infty, k)$ and $y \in (k, \infty)$ it is true that $w \in (0, \infty)$, and for a given value of w we have $x \in (k - w, k)$, so the integral becomes

$$\begin{aligned} P_k(\tau) &= \int_0^\infty \int_{k-w}^{k} \phi(x,t)p(t,x,t+\tau,x+w)dxdw \\ &= (2\pi\sigma^2\tau)^{-1/2} \int_0^\infty e^{-w^2/(2\sigma^2\tau)} \int_{k-w}^{k} \phi(x,t)dxdw \end{aligned}$$

Now we use the following approximation: $\phi(x,t)$ is constant in a interval of order $\sqrt{\sigma^2\tau}$ around x, corresponding to the range in which $e^{-w^2/(2\sigma^2\tau)}$ is different from zero. Solving the well known gaussian integral, $P_k(\tau)$ becomes

$$\begin{aligned} P_k(\tau) &\simeq \phi(k,t)(2\pi\sigma^2\tau)^{-1/2} \int_0^\infty we^{-w^2/(2\sigma^2\tau)}dw \\ &= \phi(k,t)\sqrt{\frac{\sigma^2\tau}{2\pi}} \end{aligned}$$

From this result we also have $P_k(2\tau) \simeq \sqrt{2}P_k(\tau)$.

1.5 Pykov: a python module for Markov chains

Pykov is Python module for the creation and manipulation of finite Markov chains. It is possible to define a Markov chain from scratch or read it from a text file according specific format. Pykov is versatile, being it able to manipulate the chain, inserting and removing nodes, and to calculate various kind of quantities, like the steady state distribution, mean first passage times, random walks, absorbing times, mixing times, Kemeny constant, and so on. Pykov is also really fast, being it based on Pysparse, a sparse matrix library for Python.

Pykov is licensed under the terms of the GNU General Public License as published by the Free Software Foundation.

The pykov documentation is presented as appendix at the end of the thesis, it is copied from the official webpage (http://riccardoscalco.github.com/Pykov/index.html).

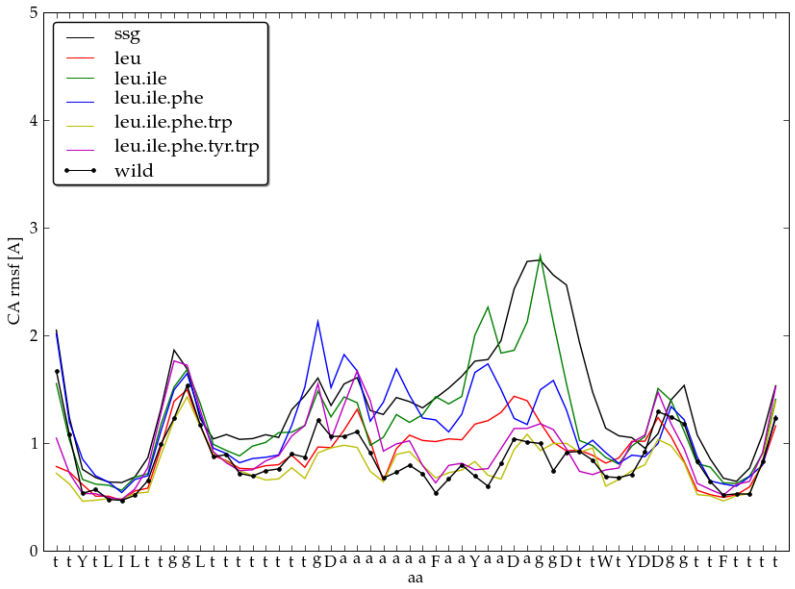


Figure 1.7: Root mean square fluctuations (RMSF) of the native conformation of the different mutants. The RMSF decrease by increasing the sequence complexity, i.e. by increasing the amino acid alphabet.

1.6 Work in progress: Simplified proteins

During my first year of PhD I studied a simplified version of the sequence of a 56-residue α/β structure (the immunoglobulin-binding domain of protein G), the idea is to investigate the native state and folding mechanisms of putatively primordial proteins. Chapter 4 introduced the analysis and the main results obtained up to now, which state that the flexibility of the folded state anticorrelates with the complexity of the sequence: higher the complexity of the sequence the more rigid is the native structure (figure 1.7).

Bibliography

[1] J. G. Kemeny and J. L. Snell (1960) Finite Markov Chains. *Springer-Verlag New York*

[2] A. R. Leach (2001) Molecular Modelling, Principles and Applications. Second Edition. *Pearson Education Limited England*

[3] Y. A. Rozanov (1982) Introduction to Random Processes. *Springer-Verlag New York*

[4] Sergei V. Krivov, Martin Karplus (2004) Hidden complexity of free energy surfaces for peptide (protein) folding. *Proc Natl Acad Sci USA* 101:14766-14770.

[5] Sergei V. Krivov, Martin Karplus (2008) Diffusive reaction dynamics on invariant free energy profiles. *Proc Natl Acad Sci USA* 105:13841-13846.

[6] Jan-Hendrik Prinz et al. (2011) Markov models of molecular kinetics: Generation and validation. *J Chem Phys* 134:174105.

Chapter 2

Equilibrium distribution from distributed computing (simulations of protein folding)

Scalco, R.; Caflisch A. *J. Phys. Chem. B* **2011**, *115 (19)*, 6358–6365

THE JOURNAL OF PHYSICAL CHEMISTRY B

ARTICLE

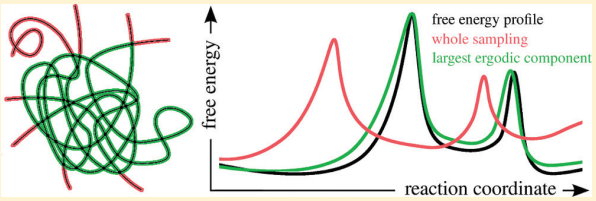
pubs.acs.org/JPCB

Equilibrium Distribution from Distributed Computing (Simulations of Protein Folding)

Riccardo Scalco and Amedeo Caflisch*

Department of Biochemistry, University of Zurich, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland

ABSTRACT: Multiple independent molecular dynamics (MD) simulations are often carried out starting from a single protein structure or a set of conformations that do not correspond to a thermodynamic ensemble. Therefore, a significant statistical bias is usually present in the Markov state model generated by simply combining the whole MD sampling into a network whose nodes and links are clusters of snapshots and transitions between them, respectively. Here, we introduce a depth-first search algorithm to extract from the whole conformation space network the largest ergodic component, i.e., the subset of nodes of the network whose transition matrix corresponds to an ergodic Markov chain. For multiple short MD simulations of a globular protein (as in distributed computing), the steady state, i.e., stationary distribution determined using the largest ergodic component, yields more accurate free energy profiles and mean first passage times than the original network or the ergodic network obtained by imposing detailed balance by means of symmetrization of the transition counts.



I. INTRODUCTION

Atomistic molecular dynamics (MD) simulations are widely used for Boltzmann-weighted (i.e., equilibrium) sampling of the phase space of biological macromolecules.[1] In principle, MD simulations of length significantly longer than the process of interest should generate a molecular "movie" at very high spatial and temporal resolution. In practice, because of the many degrees of freedom in the (poly)peptide chain and the related complexity of the free energy surface it is very challenging to sample the conformational space of proteins and even peptides by standard MD techniques, which have an inherently "slow" time step of about 1−5 fs. At low temperatures, MD simulations can get trapped in the starting basin. At elevated temperatures, on the other hand, the accessible phase space increases enormously so that not all possible conformations are visited. A number of simulation techniques have been introduced to enhance the sampling of the conformational space.[2−8] At the same time, the availability of hundreds to thousands of processors has been exploited by intrinsically parallel jobs like distributed computing[9,10] and loosely coupled MD simulations.[11] Because of the significant time-scale gap between the actual protein folding process (microseconds to seconds) and simulation length (nanoseconds), it is not possible to extract folding kinetics *directly* from distributed computing simulations.[10,12]

Markov chain models have been used to determine transition probabilities between coarse-grained states. These states (or more precisely clusters of MD snapshots) usually range in number between 100 and 1000, and have been derived from multiple short MD runs[13−16] or from long trajectories with multiple folding and unfolding events.[17] One advantage of Markov state models is that they can be used to combine (short) independent MD simulations for extracting information on processes whose time-scale is longer than the one of the individual MD runs.[13,18−20] A potential disadvantage is that the sampling of phase space by multiple, independent short runs can be affected by a statistical bias.[21] Such bias is easily understood considering that the starting nodes are selected following a probability distribution which is different from the Boltzmann-weighted distribution. Under the assumption that the transition probabilities between coarse-grained states, which are conditional probabilities of local transitions, are sampled correctly, the bias can be removed by calculating the steady state of the Markov chain. For the steady state calculation, the Markov chain must be ergodic,[22] in other words it must be irreducible (from any state the system can reach every other state) and aperiodic (there are no states which show up at a fixed period of time). Markov chains derived from multiple MD trajectories are usually not ergodic. The nonergodicity is a consequence of the sampling by multiple (short) runs, e.g., most initial and final conformations act as sources and sinks, respectively, which make the Markov chain nonirreducible.

Here we show that an automatic procedure for the identification of the largest ergodic component from a nonirreducible directed network (shown schematically in Figure 1) is able to remove the statistical bias which is typical of MD sampling by multiple short trajectories. The procedure used here is based on a theorem[23] that expresses the possibility to subdivide every directed graph (the terms "network" and "graph" are used as synonyms) in its irreducible components, the largest of which is likely to be the most relevant. The method has several advantages: first, it does not require any parameter; second, it translates

Received: February 15, 2011
Revised: April 8, 2011
Published: April 25, 2011

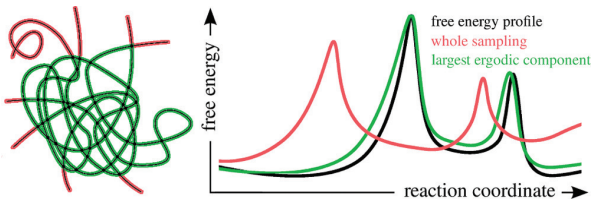


Figure 1. Schematic illustration of the procedure described in this paper. Left: The largest subset of nodes that yields an ergodic network (green) is extracted from the whole sampling (red and green). In this way, it is possible to remove the statistical bias, which is usually introduced by multiple independent trajectories. Right: The free energy profile of the largest ergodic component (green) is much closer to the actual profile (black) than the profile obtained considering the whole sampling (red).

the problem of obtaining an irreducible network in the search for the minimum number of links to be removed to obtain irreducibility; third, several computational implementations already exist. Here we use the algorithm presented by Tarjan,[23] which makes use of a *depth-first search* in the graph and thus is very efficient.

This paper is structured as follows. The Theory section presents the principles of Markov state models derived from MD sampling, the procedure for the extraction of the largest ergodic component, and an analytical formula of the dependence of the statistical bias on the number of simulations, their length, and the fundamental matrix associated with the transition matrix. In the Examples section, the kinetics obtained by extraction of the largest ergodic component are compared with the broadly used imposition of detailed balance by simple symmetrization of the matrix of transition counts. It is shown that for multiple short trajectories, the statistical bias cannot be removed by imposing detailed balance which results in wrong statistics. In contrast, the largest ergodic component yields free energy profiles and mean first passage times that better reflect the kinetics than the results obtained by detailed balance imposition. The Conclusion summarizes the main points of this work.

II. THEORY

A. Markov State Models from Multiple MD Simulations. We consider the frequent case of m independent molecular dynamics or Metropolis Monte Carlo runs for which it is convenient to introduce the abbreviation m-trj instead of the more generic term *trajectory*. More precisely, with m-trj we intend the m symbolic trajectories obtained usually by a clustering procedure of the whole sampling.[24]

We assume that the system being studied is ergodic and can be formalized as a finite homogeneous Markov chain. Thanks to the ergodic hypothesis, we can use Birkhoff's theorem[25] to extract from the m-trj the transition matrix P associated with the Markov chain. If the m-trj is long enough, the transition probabilities will converge. Here, we assume to obtain such convergence, or at least to obtain it in a certain subspace of interest of the system phase space. Indeed such transition probabilities are local, i.e., they are conditional probabilities and therefore not affected by an incomplete sampling of the phase space.

Given an m-trj, we use *naïve* definitions, namely the maximum likelihood estimates,[21] for the probability distribution p_i over the nodes set $\{i\}$, the transition probabilities P_{ij} between nodes, and the transition rates q_{ij}. Let start by defining q_{ij} as the number of one step transitions $i \rightarrow j$ observed in the m-trj, normalized by the total number of transitions. From q_{ij} we derive $p_i = \Sigma_j q_{ij}$, so that $\Sigma_i p_i = 1$. Finally, the transition probabilities are the conditional probabilities $P_{ij} = q_{ij}/p_i$. Note that, if $p_i > 0$ for every node i, P is by definition a right stochastic matrix, i.e., a square matrix whose rows consist each of non-negative real numbers that sum up to 1: $\Sigma_j P_{ij} = \Sigma_j (q_{ij})/(p_i) = 1$.

In general, given an m-trj, the Markov chain P obtained with the above definitions is not ergodic. Because of the finite sampling, one or more segments of the m-trj act as attractor(s) so that the directed graph associated with the chain is not irreducible. We stress that the non irreducibility of the chain could be a solvable problem which is easily fixed by considering that such attractors are contained in the statistically less informative part of the sampling, so that they can be discarded without any significant loss of statistics. In essence, we are concerned with the statistical bias of which every m-trj is affected. In other words, the choice of starting point(s) of the m 1-trj does not reflect the correct probability distribution over the nodes. To remove such bias, we need an equilibration procedure, namely we calculate the steady state π of the Markov chain, the state that satisfies the equation $\pi = \pi P$.[22] Only ergodic chains possess one and only one steady state, the entries of which are all positive. Thus, we need to retrieve an ergodic chain from the m-trj before calculating the steady state.

B. The Largest Ergodic Component. The problem of how to obtain an irreducible graph from a nonirreducible one does not have a straightforward solution, because the problem itself is not well-defined. In other words, given a directed graph, many different irreducible graphs can be generated from it. A common solution, due to its simplicity, is to impose detailed balance by symmetrizing the count matrix, i.e., defining $q_{ij}^{db} = q_{ji}^{db} = (q_{ij} + q_{ji})/2$, which corresponds to including the counts that would have been obtained by the time-reversed simulations.[21,26] The symmetrization of the count matrix introduces an error which is larger the larger the difference between the actual sampling and the equilibrium sampling is (see below). In other words, imposing detailed balance directly to the nonergodic graph renders impossible the removal of the statistical bias connected to the m-trj. We suggest to obtain ergodicity from the collected transitions without modifying their statistical nature, i.e., we advise against the insertion of spurious transitions as in the symmetrization of the count matrix. Instead, we suggest *to remove the minimum amount of links to obtain an irreducible directed graph*. We prefer to remove instead of insert links in order to affect the statistics as little as possible.

With the above task in mind, i.e., removing the minimum amount of links for generating an ergodic graph, we make use of following graph theory theorem.[27] Given a directed graph $G = (V,E)$, where V and E are the sets of vertices and edges, it is possible to define an equivalence relation on V such that two vertices v and w are equivalent if there is a path from v to w and a path from w to v. Let V_i, i:1, ..., n, the n distinct equivalence classes, defining $G_i = (V_i,E_i)$, where $E_i = \{(v,w) \in E | v,w \in V_i\}$, one can prove that

- each G_i is strongly connected (irreducible)
- no G_i is a proper subgraph of a strongly connected subgraph of G (a proper subgraph of G is a subgraph which contains at least one and not all the edges of G)

The subgraphs G_i are called the strongly connected (or irreducible) components of G. We note that the subdivision in equivalence classes is unique, so we do not need any parameter to obtain the strongly connected components. Moreover, the condition of minimal removal of links is satisfied by the second point, namely the subgraphs G_i are the largest possible components.

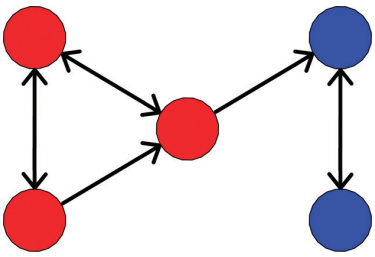


Figure 2. Example of a directed graph with two strongly connected components emphasized by different colors.

Figure 2 shows an example of a directed graph with two strongly connected components.

Hopefully, the largest strongly connected component covers the most relevant part of the original graph. It will be shown in the results that the larger is the sampling, the smaller is the number of links to be removed to obtain the largest strongly connected component. Assuming the largest strongly connected component is aperiodic, as is always the case for complex networks describing free energy surfaces of peptides and proteins,[24,28] it is also the largest ergodic component. Finally, we emphasize that such procedure is not a community detection algorithm,[29−31] it simply solves the well-defined mathematical problem regarding the subdivision in strongly connected components of a generic directed graph.

The theorem enunciated above has been employed in different computer algorithms. Here, the algorithm published by Tarjan is used.[23] It is based on a *depth-first search* procedure in the graph which is very efficient. For the largest network in the examples mentioned below (network of example A with 3652 nodes and 18948 links) the Tarjan algorithm requires less than one second on a 3 GHz commodity processor. In general, it requires $O(V,E)$ space and time, namely the algorithm needs space bounded by $k_1V + k_2E + k_3$, where k_1, k_2, and k_3 are constant.

C. The Statistical Bias. To illustrate the origin of the bias, it is useful to formulate an analytical formula of the deviation from the stationary distribution. In the following, P is the transition matrix associated with an ergodic Markov chain C. We remember that from any m-trj it is possible to generate an ergodic chain C by means of the Tarjan algorithm, which, as mentioned above, extracts the strongly connected components from the directed graph drawn following the m-trj. The matrix P we are looking for will be the one associated with the ergodic chain C generating the largest irreducible component. In general, the chain C consists of several trajectories of different length extracted from the original m-trj, which all together draw an irreducible directed graph.

Given the chain C, we calculate the transition rates q_{ij}, the probability distribution p_i, the transition matrix P and the steady state π. The latter is the solution of $\pi = \pi P$, and in this work it is calculated iteratively by means of $p_n = p_{n-1}P$ until convergence is reached. The relevant question is: *How does the difference $\pi - p$ depend on the sampled m-trj?* This question can be formalized for each node i using the transition rates q_{ij} as follows:

$$\sum_j q_{ij} = k_i + \sum_j q_{ji}$$

Here the index runs over the nodes. The quantity k_i is the difference between the *outgoing* and *ingoing* flow of the node i and is caused by the finite length of the m 1-trj. Of course the total sum must be zero:

$$\sum_i k_i = \sum_{ij} q_{ij} - \sum_{ij} q_{ji} = 1 - 1 = 0$$

Note that for equilibrated transition rates we expect $k_i = 0\ \forall i$, i.e., the flow conservation law $\Sigma_j q_{ij} = \Sigma_j q_{ji}$. This can be easily seen by defining q_{ij} by means of the steady state π, i.e., $q_{ij}^{eq} = \pi_i P_{ij}$, so that

$$\sum_j q_{ij}^{eq} = \sum_j \pi_i P_{ij} = \pi_i \sum_j P_{ij} = \pi_i$$

$$\sum_j q_{ji}^{eq} = \sum_j \pi_j P_{ji} = [\pi P]_i = \pi_i$$

In other words, the presence of $k_i \neq 0$ requires the determination of the steady state to have an unbiased statistics.

With the naïve definitions of the probability distribution over the nodes ($p_{0_i} = \Sigma_j q_{ij}$) and the entries of the transition matrix P ($P_{ij} = q_{ij}/p_{0_i}$) derived from the chain C, we are able to prove the following equality for $p_m = p_0 P^m$:

$$\lim_{m\to\infty} p_m = p_0 - \lim_{m\to\infty} \sum_{n=0}^{m} kP^n$$

Proof. From equation $\Sigma_j q_{ij} = k_i + \Sigma_j q_{ji}$, we have

$$p_{0_i} = k_i + \sum_j p_{0_j} P_{ji} = k_i + p_{1_i}$$

or, without the index

$$p_0 = k + p_0 P = k + p_1$$

So, we can write

$$\begin{aligned} p_1 &= p_0 - k \\ p_2 &= p_1 P = p_0 P - kP = p_1 - kP = p_0 - k - kP \\ p_3 &= p_2 P = p_0 - k - kP - kP^2 \\ &\vdots \\ p_m &= p_0 - \sum_{n=0}^{m-1} kP^n \end{aligned}$$

taking the limit, given that P is ergodic, we briefly write

$$\pi \equiv p_\infty = p_0 - \sum_{n=0}^{\infty} kP^n$$

Summing up, the difference we are looking for is $\pi - p_0 = -\Sigma_{n=0}^{\infty} kP^n$.

The equation indicates that the series $\Sigma_{n=0}^{\infty} kP^n$ converges. To appreciate this point we rewrite the result in terms of the fundamental matrix[32] associated with the transition matrix P:

$$Z = \mathbb{I} + (P - P^\infty) + (P^2 - P^\infty) + \cdots = (\mathbb{I} - P + P^\infty)^{-1}$$

Observing that

$$\lim_{n\to\infty} [kP^n]_i = \sum_j k_j [P^\infty]_{ji} = \pi_i \sum_j k_j = 0$$

we can now rewrite the series in the form

$$\pi = p_0 - \sum_{n=0}^{\infty} kP^n = p_0 - kZ$$

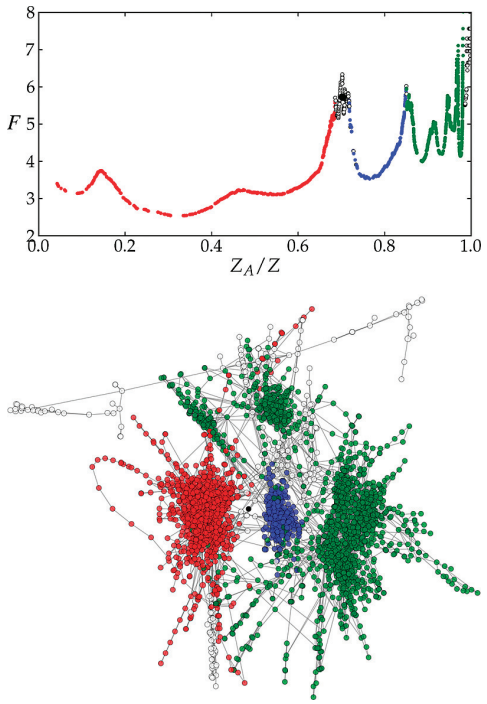


Figure 3. Free energy surface of the model system used as benchmark. The sampling was obtained by implicit solvent MD simulations of a simplified-sequence variant of protein G at 330 K[17] for which a C_α rmsd coarse-graining with 3.5 Å cutoff resulted in 3683 clusters, i.e., nodes of the network (see text). (Top) cFEP of the model. The free energy F is given in kcal/mol in all cFEPs in this work. (Bottom) The network representation was generated by the Fruchterman−Reingold force-directed algorithm.[39] The 27742 links between pairs of node represent MD transitions at 20 ps saving frequency. The coloring reflects the main basins, red and blue, which have been identified by plotting the cFEP separately from their representative node. The collection of green nodes could be furthermore divided in three smaller basins. The white nodes are unassigned, i.e., at free energy barriers. The black node at the barrier is the starting node of example D.

Moreover, we could rewrite the equality $\pi = p_0 - kZ$ separating the different factors involving the sampled m-trj. Let k_i be the difference in outgoing and ingoing transitions in the node i, m the number of 1-trj of which the chain C if formed and s the total number of transitions observed; then we note that $\Sigma_i|k_i| \leqslant 2m$ and $k_i = k_i/s$. Extracting m we define $\lambda_i \equiv k_i/m$, then we have $\Sigma_i|\lambda_i| \leq 2$ and the equality becomes $\pi = p_0 - (m/s)\lambda Z$. Thus, the difference $\pi - p_0$ depends on distinct factors:

1 The multiplicative term m/s shows that $\pi - p_0$ increases with the number m of 1-trj and decreases with the total length s of them, as expected. Note that s/m is the mean number of steps per 1-trj.
2 The vector $\lambda \equiv k/m$ depends on the initial and final nodes of the m 1-trj and it is influenced by the choice of starting nodes. For example, in the exotic case of multiple runs each of them starting and ending at the same node, λ is the null vector and there is no bias, whatever m is.
3 The shape of the visited free energy surface, which affects the fundamental matrix Z.

Some observations are needed. In the case of a m-trj consisting of few long simulations, the ratio m/s is small and, as expected, p_0 is a good approximation of π. Vice versa, for a m-trj of many short runs, the mean number of steps per simulation is small (m/s big) and the steady state calculation could be significant. Only for a choice of starting nodes that exactly follows the probability distribution at equilibrium π, the term λZ will have entries nearly equal to zero, so to make p_0 a good approximation of π. Summing up, for multiple trajectory analysis of many short runs, typical of parallel and distributed computing, that start always from the same conformation or from different conformations of unknown distribution, the steady state π offers the correct statistics.

Let us now compare π with the statistics resulting from the detailed balance imposition. The probability distribution over the nodes obtained by count symmetrization is stationary because

$$\sum_j p_j^{db} P_{ji}^{db} = \sum_j p_j^{db} \frac{q_{ji}^{db}}{p_j^{db}} = \sum_j q_{ij}^{db} = p_i^{db}$$

The stationary probability vector p_i^{db} differs from the stationary distribution π as follows:

$$p_i^{db} = \sum_j (q_{ij} + q_{ji})/2 = (p_{0_i} + p_{1_i})/2 = p_{0_i} - k_i/2$$

$$= \pi_i + [kZ]_i - k_i/2$$

It is crucial to note here that the difference between p_i^{db} and π strongly depends on the sampled m-trj. Thus, there are situations for which imposing detailed balance by simple count symmetrization is not appropriate, particularly when the m-trj consists of many short runs like in parallel and distributed computing as will be shown in the next section.

III. EXAMPLES

In this section, we illustrate the usefulness of the automatic procedure to extract the largest ergodic component, which is a subset of nodes whose transition matrix corresponds to an ergodic Markov chain. The protein used is a simplified-sequence variant of protein G[17] which is sampled by implicit solvent[33] MD at 330 K. First, the ~220000 snapshots saved every 20 ps along the MD simulations are clustered by C_α rmsd and a threshold of 3.5 Å using the leader-algorithm as implemented in WORDOM.[34,35] The clustering yields 3683 nodes, and there are 27742 links between them. The transition matrix associated with the 3683 clusters is ergodic as detailed balance condition was imposed. This transition matrix and associated stationary distribution are referred to as the "model" in the following. The cut-based free energy profile (cFEP)[36] and conformational space network[24] of the model are shown in Figure 3.

The transition matrix of the model is used to generate the m-trj sampling, i.e., to propagate m (short) trajectories of a random walker, which emulate m independent MD runs. Every step of the random walker represents a time interval of 20 ps because of the saving frequency of the MD simulations from which the network is extracted. Four examples of m-trj are discussed. They differ in the choice of the starting node(s), the number of random walker trajectories m, and/or the length $l = s/m$ of each trajectory (see Table 1 for details). Using the naïve definitions it is straightforward to determine $[P^{db}]_{ij} \equiv q_{ij}^{db}/p_i^{db}$ which is the transition matrix derived imposing detailed balance by symmetrization of the count matrix. The transition matrix derived from the chain C associated with the largest ergodic component is $[P^C]_{ij} \equiv q_{ij}^C/p_i^C$ where q_{ij}^C is the number of one step transitions $i \rightarrow j$ observed in

Table 1. Examples of *m*-trj Analysis[a]

	sampling				% visited phase space[b]		
example	*m*	*l*	(μs)	starting node	total	in largest erg. comp.	% discarded sampling[c]
A	10 000	10	2	random	99 (3652)	80 (2791)	18
B	1000	500	10	most pop.	86 (1868)	75 (1664)	1
C	1000	200	4	most pop.	73 (1356)	68 (1283)	0.5
D	1000	200	4	at the barrier	78 (1940)	76 (1817)	0.001

[a] The first five entries of each row list the name of the example, the number of random walkers *m* (i.e., number of emulated MD runs), the length of each run *l*, the total sampling, and the starting conformation(s), respectively. Note that the starting conformation of each of the 10000 runs of example A is drawn randomly from the 3683 nodes of the model, whereas it is a single node for examples B, C, and D. [b] The visited phase space is the sum of the state probabilities of the model over the states visited by the *m*-trj. The number of nodes visited is in parentheses. [c] The discarded sampling is the percentage of random walker steps that is not included in the largest ergodic component. For examples B, C, and D it is much smaller than the difference of the values in the two preceding columns because the discarded sampling concerns a part of phase space not sampled enough to reach its correct probability.

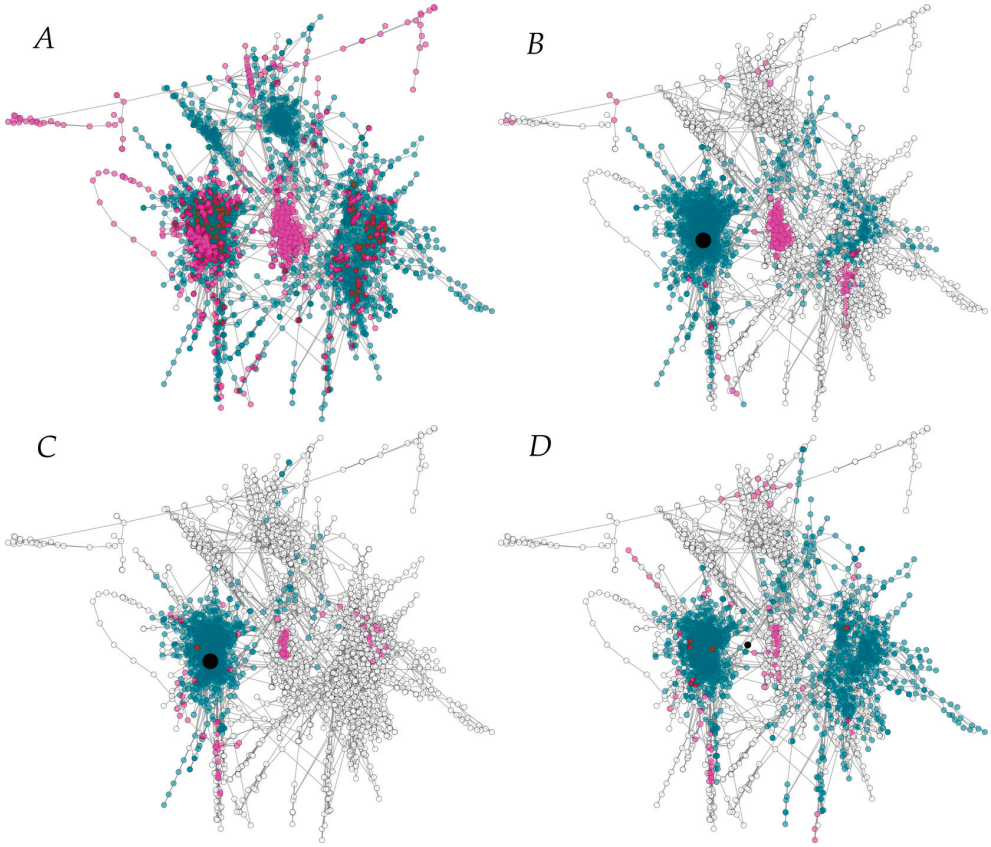


Figure 4. Network representation of the sampling in each of the four examples A to D. The nodes visited by the *m*-trj are in cyan or magenta if they are inside or outside the largest ergodic component, respectively. The white nodes were not visited by the *m*-trj and the black node is the starting node except for example A which used almost all nodes as starting nodes. The details of the four examples are given in Table 1.

the chain *C* and $p_i^C = \Sigma_j q_{ij}^C$. Therefore, $[P^{C,eq,db}]_{ij} \equiv q_{ij}^{C,eq,db}/p_i^{C,eq,db}$ is the transition matrix derived imposing detailed balance on the equilibrated transition rate probabilities $q_{ij}^{C,eq} = \pi_i P_{ij}^C$, where π is the steady state of the chain *C*, i.e., $\pi = \pi P^C$.

For each example, we calculate the cut-based free energy profile (cFEP)[36] using the most probable node as reference and the mean first passage time (mfpt) as progress coordinate.[37] The analysis focuses on the differences between the straightforward (but in most case inappropriate) count symmetrization (P^{db}) and the steady state of the largest ergodic component ($P^{C,eq,db}$).

A. Distributed Computing. Example A is an *m*-trj consisting of $m = 10000$ very short ($l = 10$) random walkers, which is the equivalent of 2 μs of sampling by implicit solvent MD, starting at nodes selected randomly (Table 1). Note that the 200 ps length of each walker corresponds to an explicit water MD time scale of about 2−20 ns (i.e., 10 to 100 longer[38]) because of the lack of

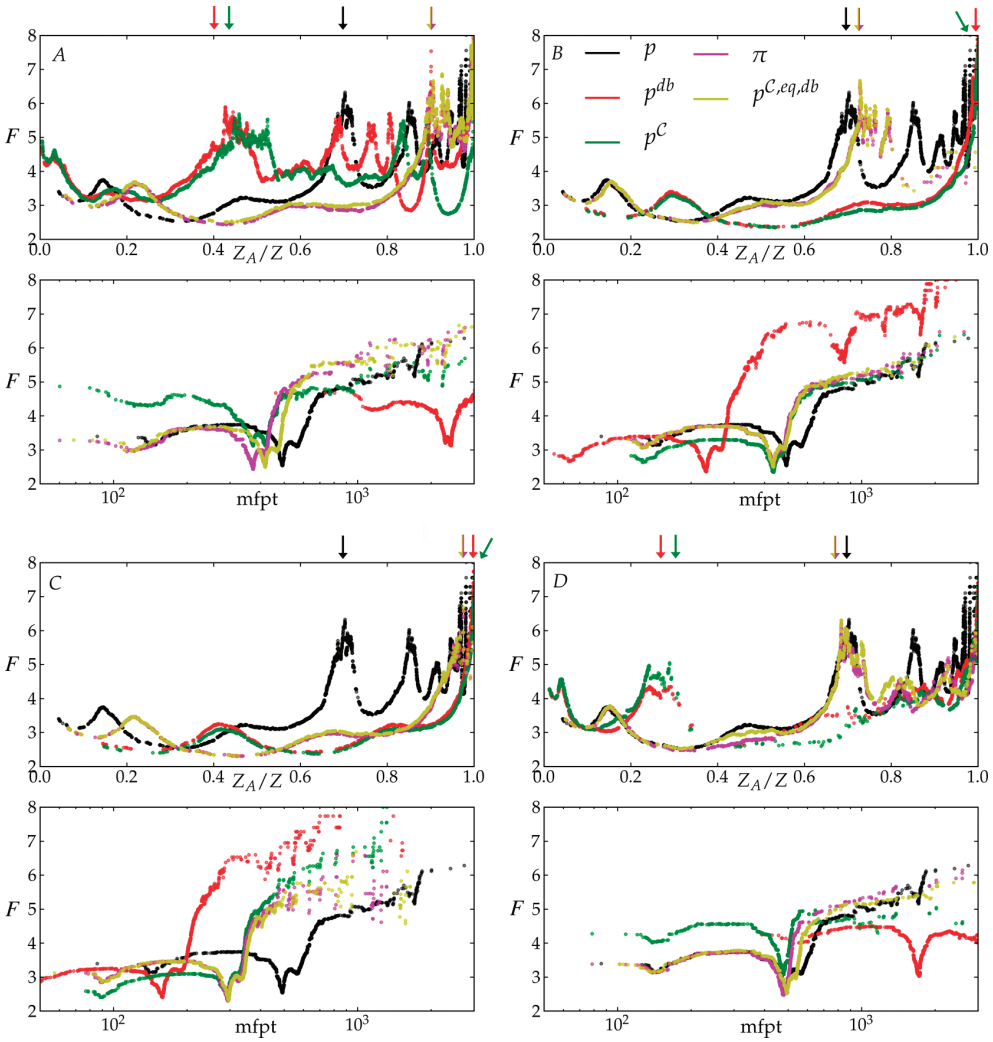


Figure 5. Free energy profiles and mfpt values of examples A, B, C, and D. The top and bottom parts of each of the four panels show the cFEP plotted using as reaction coordinate the relative partition function Z_A/Z (ref 36) and mfpt (ref 37), respectively. Note that to improve resolution the cFEPs plotted as a function of mfpt include only the range up to 3000 steps, i.e., 60 ns. The vertical arrows above the Z_A/Z cFEPs indicate the relative partition function value corresponding to mfpt =60 ns. The cFEP plotted as a function of Z_A/Z illustrates barrier heights and locations as obtained by different transition matrices, while the cFEP with mfpt as reaction coordinate allows the direct comparison of the mfpt values. As shown in the legend of panel B, individual cFEPs are colored as follows: Black for the original transition matrix P with probability distribution $p_i = \Sigma_j q_{ij}$ and transition rate probabilities q_{ij}; red for the naïve symmetrization of the transition counts, resulting in the transition matrix P^{db} with probability distribution $p_i^{db} = \Sigma_j q_{ij}^{db}$, where $q_{ij}^{db} = q_{ji}^{db} = (q_{ij} + q_{ji})/2$; green for the largest ergodic component P^C with naïve definitions of p_i^C and q_{ij}^C; magenta for P^C with the steady state π and $q_{ij}^{C,eq}$; yellow for $P^{C,eq,db}$ with $q_{ij}^{C,eq,db} = (q_{ij}^{C,eq} + q_{ji}^{C,eq})/2$ and $p_i^{C,eq,db} = \Sigma_j q_{ij}^{C,eq,db}$. The comparison of P^{db} (red) and $P^{C,eq,db}$ (yellow) is useful to analyze the statistical bias.

friction in the implicit solvent MD simulations.[17] Since the starting nodes are chosen uniformly and not according to the distribution of node size, the initial ensemble does not reflect the Boltzmann distribution, which is often the case in distributed computing.[10,12] A total of 18% of random walker steps (i.e., 18% of the *m*-trj sampling) are outside of the largest ergodic component identified by the Tarjan algorithm (in less than 1 s). A comparison of the networks colored according to the individual free energy basins (Figure 3) and according to the largest ergodic component (Figure 4A) indicates that most of the discarded sampling lies outside of the most populated basin and is located in the region of the free energy surface colored in blue in Figure 3. This part of sampling concerns the second largest irreducible component, which could be connected with the largest one by means of further sampling at the barrier between the red and the blue basins.

The symmetrization of the count matrix of the whole *m*-trj sampling yields a free energy profile very different from the model and too large mfpt values of the nodes within the most populated basin (Figure 5A, red curves). Because of the very short length of the random walker trajectories ($1/l \equiv m/s = 0.1$) and the choice of the starting nodes (shape of the λ vector), steady state calculation is expected to be necessary. Despite the 18% loss of *m*-trj sampling due to the extraction of the largest ergodic component, the transition matrices P^C with the steady state and $P^{C,eq,db}$ yield very good approximations of the main

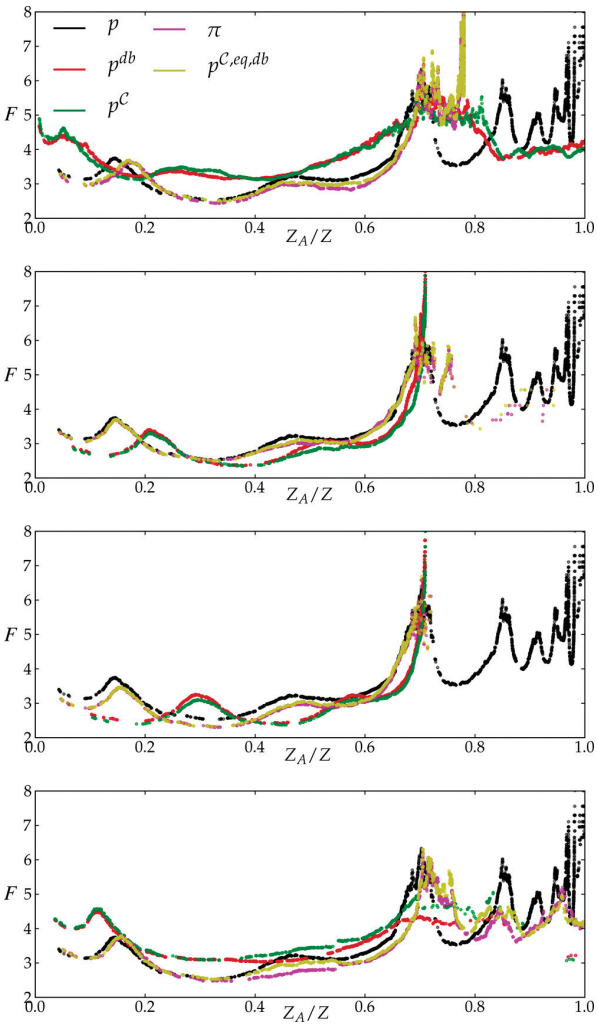


Figure 6. Stationary distribution on the largest ergodic component extracted from the *m*-trj of examples A, B, C, and D, yields the same cFEP of the main free energy basin as the original model. For a direct comparison, a rescaling of the Z_A/Z reaction coordinate is applied to the cFEP calculated using the $P^{C,eq,db}$ transition matrix. More precisely, given for the original model $p_A = \Sigma_i p_i$ for all nodes i in the main basin (whose range is $0 \leq Z_A/Z < 0.7$), and p'_A the same quantity for the other transition matrices, a coordinate transformation is applied to the x-axis by means of a rescaling factor $C = p_A/p'_A$ for the p_i values of the *m*-trj. Note that the rescaling is possible because the cFEP is invariant with respect to any continuous invertible transformation of the reaction coordinate.[40] The colors of the cFEPs are explained in the legend of Figure 5.

barrier on the cFEP and mfpt values. Moreover, their cFEPs are essentially identical. The position of the barrier on the x-axis corresponds to the statistical weight of the most populated basin. The shift of the barrier by about 18% in the cFEPs of P^C and $P^{C,eq,db}$ with respect to the original model is consistent with the result of the aforementioned network-coloring comparison, i.e., that the steps of the walkers not included in the largest ergodic component are mainly located outside of the most populated basin. Note however that the barrier height is preserved. Both of these findings are further illustrated by a transformation of the reaction coordinate (i.e., rescaling of the relative partition function) in the cFEP plot (Figure 6A).

Since a significant fraction of nodes is lost upon extraction of the largest ergodic component, it is somewhat surprising that mfpt values and cFEP profiles up to the first barrier are very accurate. There are two main reasons for these observations. First, both mfpt and cFEP are calculated using the transition probabilities and the equilibrium distribution, which are not affected by the bias introduced by multiple trajectory sampling. Second, the relaxation kinetics inside a free energy basin depends only on the profile of the basin up to the barrier to leave the basin.

B. Influence of Simulation Length. Examples B and C are *m*-trj composed of $m = 1000$ random walkers starting always at the most populated node of the model. They differ in the length of each walker trajectory which is $l = 500$ (resulting in a total of 10 μs sampling) and $l = 200$ (4 μs sampling) in examples B and C, respectively. As in example A, P^C with the steady state and $P^{C,eq,db}$ yield similar cFEPs and mfpt values. Importantly, P^C and $P^{C,eq,db}$ approximate correctly the original model in example B but not in example C (Figures 5B and 5C), which reflects that the simulation length plays an important role particularly when all runs start from the same structure. The network illustrations and cFEPs show that the $l = 500$ walkers have sufficient time to jump over the main barrier and visit other basins besides the most populated one (example B, i.e., Figures 4B and 6B) while the $l = 200$ walkers do not leave the main basin (example C, i.e., Figures 4C and 6C).

C. Influence of Starting Structure. Examples C and D are *m*-trj composed of $m = 1000$ random walkers each of $l = 200$ steps, which is the equivalent of 4 μs of sampling by implicit solvent MD. These two examples differ in the starting structure which, as mentioned above, is the most probable node of the model in example C (Figure 4C), and a very low-populated node at the top of the main barrier, i.e., the barrier to escape from the most probable basin, in example D (Figure 4D).

The stationary distribution of the largest ergodic component and the associated transition matrices P^C and $P^{C,eq,db}$ yield a much better approximation of the main barrier on the cFEP and mfpt values than P^{db} (Figure 5C,D). Moreover, the mfpt values are more accurate in example D than C which is consistent with the location of the starting node. Notably, using the stationary distribution, the sampling generated by starting at the main barrier yields the most accurate mfpt values of the four examples (compare magenta and yellow profiles with black profile in Figure 5D). In striking contrast, for the nodes within the most populated basin the simple symmetrization of the count matrix (P^{db}) in example D (Figure 5D) yields mfpt values that are significantly larger than the model, which is another indication of the error related to naïvely considering the transitions of the time-reversed simulations.

IV. CONCLUSIONS

Distributed computing and massively parallel computers have fostered the sampling of (small) protein conformational space by multiple, independent MD runs. The individual MD simulations are usually much shorter, particularly in distributed computing, than the time-scales associated with relevant conformational transitions, like protein folding and protein/protein association. As a consequence, the sampling obtained by independent MD runs is usually biased because of the short length of each run and/or the choice of the starting conformation(s).

In the present work, the statistical bias of multiple MD trajectories is formulated by an analytical expression that describes the dependence on the length of the trajectories, the choice of the starting conformation(s), and the underlying free energy surface. More precisely, an analytical formulation is given for the difference between the stationary distribution (or steady state) and the probability distribution obtained by simple symmetrization of the count matrix.

An automatic procedure is introduced for extracting the largest irreducible component from the whole conformational space network, or more precisely, the largest subset of nodes of the network whose associated transition matrix reflects an ergodic Markov chain. From the latter, the stationary distribution can be determined and used for calculating mfpt values and cFEP. The algorithm by Tarjan for the determination of the irreducible components is very efficient (linear on the number of nodes and links). Its application to four examples of MD sampling by multiple short trajectories shows that the stationary distribution on the largest ergodic component of the original network yields more accurate mfpt values and cFEPs than the naïve symmetrization of the count matrix. Thus, Tarjan's algorithm could be combined to network and cFEP analysis to search for weakly sampled regions of conformational space between two (or more) strongly connected components. This information could be very useful for improving an initial sampling by further MD simulations.

The automatic procedures for extracting the largest ergodic component and for determining the stationary distribution have been implemented in WORDOM.[35]

AUTHOR INFORMATION

Corresponding Author

*Telephone: +41 44 635 55 21. Fax: +41 44 635 68 62. E-mail: caflisch@bioc.uzh.ch.

ACKNOWLEDGMENT

We thank Dr. Andreas Vitalis for interesting discussions and comments to the manuscript. We thank Dr. Michele Seeber for the WORDOM implementation of the procedure for extraction of the largest ergodic component. This work was supported by a Swiss National Science Foundation grant to A.C.

REFERENCES

(1) McCammon, J. A.; Karplus, M. *Nat. Struct. Biol.* **2002**, *9*, 646–652.
(2) Krivov, S. V.; Chekmarev, S. F.; Karplus, M. *Phys. Rev. Lett.* **2002**, *88*, 038101.
(3) Dickson, A.; Dinner, A. R. *Annu. Rev. Phys. Chem.* **2010**, *61*, 441–459.
(4) Okamoto, Y. *J. Mol. Graph. Model* **2004**, *22*, 425–439.
(5) Melchionna, S. *Phys. Rev. E* **2000**, *62*, 8762–8767.
(6) Bonomi, M.; Parrinello, M. *Phys. Rev. Lett.* **2010**, *104*, 190601.
(7) Zhang, C.; Ma, J. *J. Chem. Phys.* **2010**, *132*, 244101.
(8) Muff, S.; Caflisch, A. *J. Phys. Chem. B* **2009**, *113*, 3218–3226.
(9) Snow, C. D.; Nguyen, H.; Pande, V. S.; Gruebele, M. *Nature* **2002**, *420*, 102–106.
(10) Paci, E.; Cavalli, A.; Vendruscolo, M.; Caflisch, A. *Proc. Natl. Acad. Sci. U.S.A.* **2003**, *100*, 8217–8222.
(11) Settanni, G.; Gsponer, J.; Caflisch, A. *Biophys. J.* **2004**, *86*, 1691–1701.
(12) Fersht, A. R. *Proc. Natl. Acad. Sci. U.S.A.* **2002**, *99*, 14122–14125.
(13) Pitera, J. W.; Swope, W. C.; Suits, F. *J. Phys. Chem. B* **2004**, *108*, 6571–6581.
(14) Singhal, N.; Snow, C. D.; Pande, V. S. *J. Chem. Phys.* **2004**, *121*, 415–425.
(15) Chodera, J. D.; Singhal, N.; Pande, V. S.; Dill, K. A.; Swope, W. C. *J. Chem. Phys.* **2007**, *126*, 155101.
(16) Noé, F. *J. Chem. Phys.* **2008**, *128*, 244103.
(17) Guarnera, E.; Pellarin, R.; Caflisch, A. *Biophys. J.* **2009**, *97*, 1737–1746.
(18) Pitera, J. W.; Chodera, J. D.; Swope, W. C.; Dill, K. A. *Multiscale Model. Simul.* **2006**, *5*, 1214–1226.
(19) Sriraman, S.; Kevrekidis, I. G.; Hummer, G. *J. Phys. Chem. B* **2005**, *109*, 6479–6484.
(20) Yang, S.; Banavali, N. K.; Roux, B. *Proc. Natl. Acad. Sci. U.S.A.* **2009**, *106*, 3776–3781.
(21) Bowman, G. R.; Beauchamp, K. A.; Boxer, G.; Pande, V. S. *J. Chem. Phys.* **2009**, *131*, 124101.
(22) Kemeny, J. G.; Snell, J. L. *Finite Markov Chains*; Undergraduate Texts in Mathematics; Springer-Verlag: New York, 1976.
(23) Tarjan, R. *SIAM J. Comput.* **1972**, *1*, 146–160.
(24) Rao, F.; Caflisch, A. *J. Mol. Biol.* **2004**, *342*, 299–306.
(25) Khinchin, A. I. *Mathematical Foundations of Information Theory*; Dover: New York, 1957.
(26) Muff, S.; Caflisch, A. *J. Chem. Phys.* **2009**, *130*, 125104.
(27) Bondy, A.; Murty, U. S. R. *Graph Theory*; Graduate Texts in Mathematics; Springer: New York, 2010.
(28) Caflisch, A. *Curr. Opin. Struct. Biol.* **2006**, *16*, 71–78.
(29) Fortunato, S.; Lancichinetti., A. *Phys. Rev. E* **2009**, *80*, 56117.
(30) Fortunato, S. *Phys. Rep.* **2010**, *486*, 75–174.
(31) Schuetz, P.; Caflisch, A. *Phys. Rev. E* **2008**, *78*, 26112.
(32) Grinstead, C, M.; Snell, J. L. *Introduction to Probability*, 2nd ed.; American Mathematical Society: Providence, RI, 1998.
(33) Ferrara, P.; Apostolakis, J.; Caflisch, A. *Proteins* **2002**, *46*, 24–33.
(34) Seeber, M.; Cecchini, M.; Rao, F.; Settanni, G.; Caflisch, A. *Bioinformatics* **2007**, *23*, 2625–2627.
(35) Seeber, M.; Felline, A.; Raimondi, G.; Muff, S.; Friedman, R.; Rao, F.; Caflisch, A.; Fanelli, F. *J. Comput. Chem.* **2011**, *32*, 1183–1194.
(36) Krivov, S. V.; Karplus, M. *J. Phys. Chem. B* **2006**, *110*, 12689–12698.
(37) Krivov, S. V.; Muff, S.; Caflisch, A.; Karplus, M. *J. Phys. Chem. B* **2008**, *112*, 8701–8714.
(38) Cavalli, A.; Ferrara, P.; Caflisch, A. *Proteins* **2002**, *47*, 305–314.
(39) Reingold, E. M.; Fruchterman, T. M. J. *Softw. Pract. Exper.* **1991**, *21*, 1129–1164.
(40) Krivov, S. V.; Karplus, M. *Proc. Natl. Acad. Sci. U.S.A.* **2008**, *105*, 13841–13846.

Chapter 3

Ultrametricity in Protein Folding Dynamics

Scalco, R.; Caflisch A. *J. Chem. Theory Comput.* **2012**, *8 (5)*, 1580–1588

JCTC Journal of Chemical Theory and Computation

Article

pubs.acs.org/JCTC

Ultrametricity in Protein Folding Dynamics

Riccardo Scalco and Amedeo Caflisch*

Department of Biochemistry, University of Zurich, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland

ABSTRACT: The free energy of the transition state (TS) between two nodes of an ergodic Markov state model (MSM) can be obtained from the minimum cut, which is the set of edges that has the smallest sum of the flow capacities among all the possible cuts separating the two nodes. Here, we first show that the free energy of the TS is an ultrametric distance. The ultrametric property offers a way to simplify the MSM in a small number of states and, as a consequence, meaningful rate constants (free energy barriers) for the simplified MSM can be defined. We also present a new definition of the cut-based free energy profile (cbFEP), which is useful to check for the existence of a state for which the equilibration is much faster than the time to escape from it. From our analysis, a parallelism emerges between the minimum cut (maximum flow), and transition state theory (TST) or Kramers' theory.

INTRODUCTION

An N-state kinetic process can be formalized with a system of N rate equations—namely, the master equations—describing the system as a random process governed by the exponential distribution.[1,2] The N^2 rate constants appearing in the equations are supposed to be known, and the physics behind them is described by mainly two theories: transition state theory (TST)[3] and Kramers' theory.[4] Both of them define the rate constants from assumptions regarding the dynamics of the process. Being not the same theory, TST and Kramers' theory use different definitions, but it is crucial for the objective of our work to note that the two theories share some assumptions and that, in both of them, the definition of rate constant satisfies certain properties. In particular, we focus our attention on the meaning of "state" and the formal definition of the rate constant. TST has its origin in statistical mechanics, and with "state", it means a region of configuration space, namely, a subset of spatial coordinates usually in the neighborhood of an energy potential minimum. For example, in a two-state process one state is the *native state*, the other is the so-called *unfolded state*.[1] The former is described as a set of configurations around a potential minimum, while the latter includes all of the remaining configurations. In Kramers' theory, since the original paper on the diffusion model of chemical reactions,[5] the model consists of a classical particle (namely, the reaction coordinate) trapped in a one-dimensional potential well and subjected to a frictional force. Kramers asked for the rate of escape of the particle from the well. Hence, in both these theories, the term "state" indicates a finite region of the configuration space in the neighborhood of an energy minimum.

Assuming to divide the system in two states A and B, where B contains all of the phase space not occupied by A (Figure 1), in both TST and Kramers' theory, the definition of the rate constant $k_{A,B}$, between the initial state A and the final state B, is defined as the ratio $k_{A,B} \equiv Z_{A,B}/Z_A$, where Z_A is the partition function of the initial state. The quantity $Z_{A,B}$ depends only on the boundary dividing the states A and B but not on the direction; that is, $Z_{A,B} = Z_{B,A}$. TST and Kramers' theory differ from each other in the physical interpretation of the quantity $Z_{A,B}$. TST introduces the existence of a *transition state* (TS) between states A and B, and defines $Z_{A,B}$ as the partition function of the TS. Kramers' theory instead characterizes the rate $k_{A,B}$ to escape from state A by the *flux* of particles that pass through the bottleneck separating A and B.[4,5] Many approaches have been developed to calculate the flux;[4] one of them involves the calculation of the average time that the system needs to leave the domain of attraction for the first time. The key points here are the general properties of $Z_{A,B}$, namely, the dependence only on the boundary region and its independence on the direction, and the physical interpretations of it.

In describing the kinetics of protein folding by means of ergodic Markov state models (MSMs),[6−12] a natural definition of "state" and of "rate constant" emerges, and it is the objective of the present work to show how these two concepts could be defined from a cut-based free-energy analysis of the MSM.

To divide the phase space in states, it is most appropriate to identify the cut that maximizes the free energy of the TS between two nodes. Such a choice is of wide range applicability, indeed proving to be useful in protein folding dynamics[13] and in spin models of magnetic domains.[14] We show here that the free energy of the TS defines an ultrametric distance, which results in an automatic procedure to reduce the MSM, usually consisting of thousands of nodes, into few states. The proposed procedure could be seen as a community algorithm optimized for networks describing the protein folding free-energy landscape, because it is based on the definition of the free energy of the TS, i.e., the kinetics of the process. The procedure results in a reduced MSM whose states are a collection of nodes belonging to the original MSM. The number of transitions between the states (of the reduced MSM) are then used to calculate the free energy of the TS and the activation free energy between them, similar to a previous approach based on transitions observed during molecular dynamics.[15] The main goal of the present paper is not to solve problems such as lack of sampling or time scales overlapping; instead, our purpose is

Received: January 5, 2012
Published: March 10, 2012

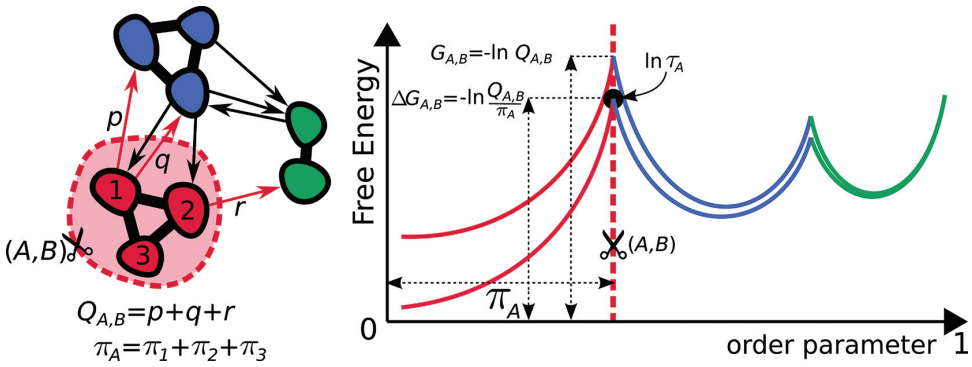


Figure 1. Schematic illustration of the main concepts. (Left) Schematic representation of a MSM consisting of eight nodes grouped in three states emphasized by different colors. The cut (A,B) (labeled by the scissor symbol) separates state A, of probability π_A, from the rest, and the cut value $Q_{A,B}$ is the flow through the cut (A,B). (Right) Graphical representation of the cut-based free energy profile (cbFEP) related to the MSM on the left. The cut (A,B) separates the profiles at the first peak, where the values $G_{A,B}$, $\Delta G_{A,B}$, and τ_A indicate the free energy of the TS, the activation free energy, and the mean time to escape from state A, respectively.

to extend the language of the mincut maxflow method applied to the kinetics of conformational transitions. In particular, we prove that, in the framework of finite Markov chains, the cut-based free energy of the TS defines an ultrametric distance on the set of states of a generic ergodic MSM that does not need to satisfy the detailed balance condition. Moreover, here, we present an alternative definition of the cut-based free-energy profile (cbFEP)[16] useful to check for the existence of a metastable state, being it easily comparable with other analysis of MSMs, such as mean exit time and mixing time.

Given the minimum cut $(A,B|i,j)$ between nodes i and j, the cut-based quantity $Q_{A,B}$ (see below) naturally fulfills both the mathematical properties and the physical interpretations of the above-mentioned quantity $Z_{A,B}$. By definition, $Q_{A,B}$ is dependent only on the cut (A,B), and, thus, $Q_{A,B} = Q_{B,A}$ represents both the flow through the cut and the partition function used to calculate the *free energy of the TS* $G_{A,B}$. It is straightforward to define the rate constant as

$$k_{A,B} \equiv \frac{Q_{A,B}}{\pi_A}$$

where π_A is the probability of finding the system in state A ($\pi_A \equiv \sum_{i\in A}\pi_i$). We will show that such definition is meaningful only if the system satisfies restricted conditions, namely, the mean time to escape from a state must be much longer than the time needed to equilibrate inside the state, and we present a way to check the validity of this assumption. Then, in similarity with the Van't Hoff–Arrhenius law, the *activation free energy*, $\Delta G_{A,B}$, can be defined as

$$\Delta G_{A,B} \equiv -\ln k_{A,B} = G_{A,B} - G_A$$

where G_A is the free energy of state A (defined as $G_A \equiv -\ln \pi_A$, given in units of k_BT).

We observe here that, by means of the cut-based free-energy definition, two distinct observables relating to states kinetics are derived: the free energy of the TS between the reference state and the remaining part of the phase space, and the free energy of activation in order to escape from the reference state. Note that the two quantities are different, and they become closer as the probability π_A to find the system in state A increases toward unity.

In the following sections, we present the theory behind the cut-based analysis. We show that the free energy of the TS defines an ultrametric distance between nodes, and how this property motivates the partition of the entire phase space in a few number of states. We then present the standard procedure to derive a Markov process in continuous time from the reduced MSM, obtaining, in this way, the master equation of the system. The rate constants appearing in the master equation are then compared with the mean escape time from the states in the original MSM, giving a strong criterion in order to establish the degree of approximation involved in the reduction procedure. This comparison is indeed related to the assumption of separated time scales for the system to equilibrate inside a state and to escape from that state; this assumption is at the base of both TST and Kramers' theory. Finally, we guide the reader along the entire cut-based analysis of a propaedeutic example and we conclude with an application to a MSM generated by molecular dynamics of the reversible folding of a structured peptide.

CUT FREE ENERGY AS METRIC DISTANCE

Assumptions and Definitions. Let P be the transition matrix defining an ergodic Markov chain[17] and G be the associated directed graph ($G \equiv (V,E)$, where V and E are the set of nodes and edges, respectively). Given the steady state of the chain ($\pi = \pi P$), the rate probability between nodes i and j is

$$q_{ij} = \pi_i P_{ij}$$

Ergodicity implies that the conservation law $\sum_j q_{ij} = \sum_j q_{ji}$ holds for every node i:

$$\sum_j q_{ij} = \sum_j \pi_i P_{ij} = \pi_i \sum_j P_{ij} = \pi_i$$

$$\sum_j q_{ji} = \sum_j \pi_j P_{ji} = [\pi P]_i = \pi_i$$

Any partition of the node set V in two disjoint subsets A and B ($A \cup B = V$, $A \cap B = \varnothing$ and $A,B \neq \varnothing$) defines a cut $C \equiv (A,B)$ on the graph G,[18,19] the *cut-set* of which is the subset of edges $C_{A,B} \equiv \{(i,j) \in E | i \in A, j \in B\}$. Weighting every edge $(i,j) \in E$

dx.doi.org/10.1021/ct3000052 | J. Chem. Theory Comput. 2012, 8, 1580–1588

by the value q_{ij}, the cut-value $Q_{A,B}$ is defined as the sum of the weights:

$$Q_{A,B} \equiv \sum_{C_{A,B}} q_{ij}$$

In the following, we make use also of expressions as "$Q_{A,B}$ is the flow going from A to B" or "$Q_{A,B}$ is the flow through the cut (A,B)". The *free energy of the TS between sets A and B* is defined as $G_{A,B} \equiv -\ln Q_{A,B}$.[13] As proved below, for an ergodic system, the symmetry property holds, namely, for every cut (A,B), it is true that $Q_{A,B} = Q_{B,A}$ and so $G_{A,B} = G_{B,A}$. With the objective of interpreting the argument of the logarithm in the above definition as a probability, we may say that it is the probability to observe on the Markov chain P a transition from a node in A to a node in B at a certain time, without conditional knowledge about which subset the initial node belongs to:

$$\sum_{i\in A} \pi_i \sum_{j\in B} P_{ij} = \sum_{\substack{i\in A\\ j\in B}} q_{ij} \equiv Q_{A,B}$$

Given two nodes i and j, we denote with the symbol $(A,B|i,j)$ a generic cut such that $i \in A$ and $j \in B$. $Q_{A,B|i,j}$ then indicates the flow through the cut $(A,B|i,j)$. Many such cuts are possible (as many as the number of bipartitions of V, such that i and j are not in the same subset) and we indicate with $Q^{\dagger}_{A,B|i,j}$ the minimum of the associated cut values:

$$Q^{\dagger}_{A,B|i,j} \equiv \min\{Q_{A,B|i,j}\}$$

The *free energy of the TS between two nodes* is then defined as

$$G^{\dagger}_{A,B|i,j} \equiv -\ln Q^{\dagger}_{A,B|i,j}$$

with the additional convention that $G^{\dagger}_{A,B|i,i} \equiv 0$. For the sake of brevity, we choose not to indicate the bipartition in subsets A and B and simply write $G^{\dagger}_{ij} \equiv -\ln Q^{\dagger}_{ij}$. An observation here is appropriate: no subset of nodes of V represents the TS between nodes i and j.

Ultrametricity. Here, we show that the free energy of the TS between any two nodes i and j ($G^{\dagger}_{ij}$) is an ultrametric distance on the set of nodes V. In other words, we must show that $G^{\dagger}_{ij}$ satisfies the following three conditions ($\forall i,j \in V$):

(1)

$$G^{\dagger}_{ij} = 0 \qquad \text{if and only if} \qquad i = j$$

To prove this property, it suffices that $G^{\dagger}_{ij} \neq 0$ if $i \neq j$ ($G^{\dagger}_{ii} = 0$ is true by definition). This is implied by the fact that for every cut (A,B) performed on a graph associated to a Markov chain, it is true that $C_{A,B} \neq E$, and so $Q_{A,B} \neq 1$.

(2)

$$G^{\dagger}_{ij} = G^{\dagger}_{ji}$$

The symmetry property is a consequence of the conservation law $\forall i \in V$: $\sum_j q_{ij} = \sum_j q_{ji}$, and of the fact that $G^{\dagger}_{ij}$ corresponds to the cut with the minimum flow. First, we show that, for every cut (A,B), it is true that $Q_{A,B} = Q_{B,A}$, where $Q_{B,A}$ is the cut value of (B,A). Remembering that a cut is a bipartition of V, we have

$$\sum_{i\in A} \sum_j q_{ij} = \sum_{i\in A} \left(\sum_{j\in A} q_{ij} + \sum_{j\in B} q_{ij}\right) = \sum_{\substack{i\in A\\ j\in A}} q_{ij} + \sum_{\substack{i\in A\\ j\in B}} q_{ij}$$

$$\sum_{i\in A} \sum_j q_{ji} = \sum_{i\in A} \left(\sum_{j\in A} q_{ji} + \sum_{j\in B} q_{ji}\right) = \sum_{\substack{i\in A\\ j\in A}} q_{ji} + \sum_{\substack{i\in A\\ j\in B}} q_{ji}$$

From the conservation law, it follows that $\sum_{i\in A}\sum_j q_{ij} = \sum_{i\in A}\sum_j q_{ji}$, which directly implies

$$Q_{A,B} \equiv \sum_{\substack{i\in A\\ j\in B}} q_{ij} = \sum_{\substack{i\in A\\ j\in B}} q_{ji} \equiv Q_{B,A}$$

because

$$\sum_{\substack{i\in A\\ j\in A}} q_{ij} = \sum_{\substack{i\in A\\ j\in A}} q_{ji}$$

Finally, from the above equality, if $(A,B|i,j)$ is the cut with the minimum flow from A to B, then the cut $(B,A|j,i)$ has the minimum flow in the opposite direction.

(3)

$$G^{\dagger}_{ij} \leq \max\left\{G^{\dagger}_{ik}, G^{\dagger}_{kj}\right\}$$

The strong triangle inequality is a consequence of the fact that, in defining $G^{\dagger}_{ij}$, we make use of the minimum cut value, $Q^{\dagger}_{A,B|i,j}$, in the set of all the possible ones. By the properties of the logarithm, the triangle inequality becomes the inequality $Q^{\dagger}_{ij} \geq \min\{Q^{\dagger}_{ik}, Q^{\dagger}_{kj}\}$. Given the cut $C = (A,B|i,j)$ associated with $Q^{\dagger}_{A,B|i,j}$, there are two mutually exclusive possibilities for the third node k: $k \in A$ or $k \in B$. In the case that $k \in A$, the cut C is also a cut $(A,B|k,j)$ and the associated cut value satisfies at

$$Q^{\dagger}_{ij} = Q_{A,B|k,j} \geq \min\{Q_{T,S|k,j}\} \equiv Q^{\dagger}_{kj}$$

That being so, in the case $k \in A$, the triangular inequality is equivalent to the logical function

$$Q^{\dagger}_{ik} \geq Q^{\dagger}_{kj} \qquad \text{OR} \qquad Q^{\dagger}_{ij} \geq Q^{\dagger}_{ik}$$

In order to show that the logical disjunction is true, it suffices that the arguments are not both false. If, by hypothesis, they are both false (namely, $Q^{\dagger}_{ik} < Q^{\dagger}_{kj}$ and $Q^{\dagger}_{ij} < Q^{\dagger}_{ik}$ are both true), then we have a reduction ad absurdum: $Q^{\dagger}_{ij} < Q^{\dagger}_{kj}$ negates the inequality $Q^{\dagger}_{ij} \geq Q^{\dagger}_{kj}$ already proven. This ensures the strong triangle inequality stated above in the case $k \in A$. In the case $k \in B$, the reasoning is entirely similar, and we omit the proof.

The second and third properties, namely, $G^{\dagger}_{ij} = G^{\dagger}_{ji}$ and $G^{\dagger}_{ij} \leq \max\{G^{\dagger}_{ik}, G^{\dagger}_{kj}\}$, are the formalizations of the following two observations. First, the free energy of the TS between two nodes is not dependent on the system direction. However the system goes from i to j, or from j to i, it must overcome the same free energy of the TS. Note that such property assumes

ergodicity but does not assume the detailed balance (even if detailed balance is expected to be fulfilled in equilibrium molecular dynamics[20]). Second, forcing the system to go from i to j, passing through k, cannot result in a lower free energy of the TS, regardless of the node k. In other words, the strong triangle inequality ensures that the only possible triangles are either isosceles with a small base or equilateral. This property could be better understood by remembering that, in the mathematical framework that we are using here, the TS is always a phase space region that acts like a ring (the cut) dividing the entire conformational space into two disjointed subsets. It is straightforward that the TS cannot be a *simply* connected region, and the choice to force the system to go from i to j passing through k cannot avoid crossing the minimum cut between i and j.

Applications: Disconnectivity Graphs, Reduced Markov Chains, and Escape Time. Ultrametricity is a relatively new concept in physics as well as in biology. A detailed review of its applications[21] shows how, despite their abstractness, ultrametric distances are of wide-range usability. Here, we are interested in the possibility of subdividing the entire network in subsets of nodes, called states, such that the simplified picture still depicts the original kinetics quantitatively. In particular, for free-energy projections that preserve the barriers,[16] it becomes apparent that such subdivision is not only possible, but also is very useful in order to derive a chemical master equation that is comparable with the experimental analysis.[22,23] Here, we use a well-known application of the ultrametricity, namely, the fact that, from any ultrametric set, a dendrogram can be unambiguously built. There is indeed a one-to-one relationship between an indexed hierarchy, a dendrogram with positive real values defined at each divergence, and an ultrametric set. Dendrograms based on potential energy or free energy (disconnectivity graphs) have been introduced in the past 15 years to characterize the shape of the multidimensional (free) energy surface.[13,24−27]

Yet, the ultrametric nature of the cut-based free energy of the TS has not been reported in previous works. In this context, it is important to note that, different from previous studies,[26,28] ergodicity is a fundamental assumption, whereas detailed balance is not required. Moreover, the definition of an ultrametric distance on the set of nodes is a different way to state the important properties of the minimum cut method, and it is potentially useful for further theoretical investigations in Markov theory, as well as in all the applications that use finite regular Markov chains models.

In order to define the dendrogram, the free energy of the TS between any pair of nodes must be calculated. This can be done by means of the isomorphism between the rate probabilities q_{ij} of the Markov chain and the capacities c_{ij} defined in a flow network. The task to find the minimum cut is then solved with standard methods such as the Ford−Fulkerson algorithm,[19] in order to find the minimum cut between two nodes, and the Gomory−Hu algorithm,[28] which is useful to deduce all the $V(V-1)/2$ minimum cuts after only $V-1$ flow problems have been computed. Different from what Gomory and Hu assumed, here, we do not assume detailed balance ($c_{ij} = c_{ji}$); nevertheless, the method still holds, because the symmetry property $Q_{A,B} = Q_{B,A}$ is true for every cut (A,B) in an ergodic chain. Note also that the strong triangle inequality, written in the form $Q^{\dagger}_{ij} \geq \min\{Q^{\dagger}_{ik}, Q^{\dagger}_{kj}\}$, is a necessary and sufficient condition for a generic matrix $Q^{\dagger}$ to be realizable by some flow network, as proven in ref 28. Moreover, the strong triangle inequality is easily understood considering its graphical interpretation on the dendrogram (see, for example, Figure 2). The free energy of the

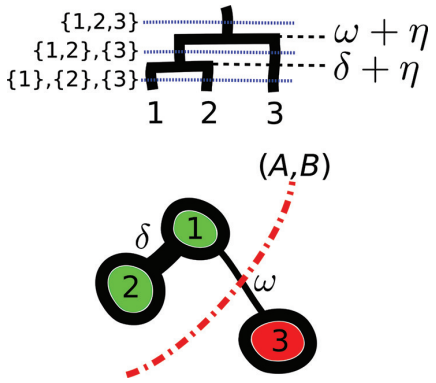


Figure 2. Depiction of a degenerate state. The top image shows the dendrogram resulting from the ultrametric distance matrix $G^{\dagger}$. Note that the free energy of the TS between two nodes is the value at the divergence between them in the hierarchical tree. The strong triangle inequality—that is, the fact that the free energy of the TS between nodes i and j cannot decrease imposing the passage through a third node k—is a consequence of the tree structure. Here, for example, it is true that $G^{\dagger}_{1,3} \leq \max\{G^{\dagger}_{1,2}, G^{\dagger}_{2,3}\}$ and $G^{\dagger}_{1,2} \leq \max\{G^{\dagger}_{1,3}, G^{\dagger}_{3,2}\}$. The bottom image shows a schematic picture of the original Markov chain subdivided into two states.

TS between two nodes i and j is the value at the divergence between them in the hierarchical tree. Therefore, the third node k must be either a descendant of the same divergence, in which case the free energy of the TS between nodes i and j passing through k does not change, or it is located outside the subtree containing nodes i and j, in which case the free energy of the TS between nodes i (or j) and k is greater than the one between i and j.

Once the dendrogram is known, a natural clustering procedure is defined by proceeding from the bottom to the top. In this way, nodes are merged into states, according to the hierarchy, and the free energy of the TS between two states corresponds to the one calculated between a node in one state and a node in the other, with the choice of these two nodes being not important.

Once the MSM is clustered in states, a reduced MSM is defined in the following way. Let x and y be two states (namely, two disjointed subsets of nodes), the rate probability between them is defined as $q_{xy} \equiv \sum_{i\in x, j\in y} q_{ij}$, and the transition probability is then calculated as $P_{xy} \equiv q_{xy}/\sum_{y} q_{xy}$. The procedure presented here to reduce a Markov chain is not free of issues; in particular, important questions emerge from the analysis of kinetic observables. Here, we focus on the following question: Is the mean escape time from a state in the reduced chain equal to the one calculated in the original chain? Generally, the answer is negative; we will determine how to check its validity using analytical calculations.

MEAN ESCAPE TIME

From a Markov chain in discrete time, it is possible to derive the corresponding Markov process in continuous time.[29,30] The Markov process is described by the master equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\pi_x(t) = \sum_y \left(k_{yx}\pi_y(t) - k_{xy}\pi_x(t)\right)$$

where the rate constants k_{xy} are the transition probabilities P_{xy} of the starting discrete time Markov chain, and the mean time to escape from a state x is equal to the inverse of the probability $\sum_{y\neq x}P_{xy} = 1 - P_{xx}$ to move from x to another state. A comparison between the mean escape time from state x of the reduced chain and the mean escape time from the subset of nodes denoting the same state x in the original (not reduced) chain is an interesting criterion in order to establish the quality of the approximation, where a good approximation requires similar values. As we will see, such a request is a necessary condition for a well-known assumption made in TST and Kramers' theory, namely, the assumption that a probability distribution of configurations belonging to a state maintain a local equilibrium form at all times. In other words, the ratio between the probabilities associated with two different configurations belonging to the same state does not change over time. By looking at the dynamic of the process, such an assumption is equivalent to ask that the content of a state must relax to equilibrium much faster than the mean time of leaving that region.[3] The role of this assumption is to neglect every deviation from thermal equilibrium distribution (namely, the Boltzmann distribution). A similar assumption is at the base of Kramers' theory: from the nonlinear dynamics of the model, a time scale separation emerges for values of the barrier height much greater than thermal energy k_BT. In that case, the random frictional force is acting as a small perturbation and the particle will have the time to equilibrate on minima of the potential well before the accumulated action of the random force will drive it over the barrier into a neighboring state. If there is no separation of time scales (that is, when the barrier height is of the order of k_BT), a rate description is not suitable.[4] Both assumptions consist of a separation between the time scale for the system to equilibrate inside a state, and the time scale to escape from it. The justification of such an assumption is generally contingent on a good partitioning of the configuration space in states.[31−34]

In this section, we present how to analyze such a separation of time scales by means of cut-based free energy. As mentioned above, in Kramers' theory, one way to calculate the rate $k_{A,B}$ consists of evaluation of the mean time τ_A to escape from state A. The equivalence of mean escape time and Kramers' rate could be easily motivated by the following reasoning. For a given ergodic Markov chain of transition matrix P, a subset A of its nodes ($B \equiv V - A$) and a probability vector v defined on nodes belonging to A, the mean time τ_A to escape from A is calculated as described in ref 17. Let P_A be the submatrix of P containing the transition probabilities of the nodes inside A, and N be the fundamental matrix of the associated absorbing Markov chain ($N \equiv (\mathrm{II} - P_A)^{-1}$); then the mean time $\tau_A(v)$ to escape from A, starting from the distribution v, is

$$\tau_A(v) = \sum_{i\in A} v_i \sum_{j\in A} N_{ij}\varepsilon_j$$

where ε is the column vector with all entries equal to 1. The vector $Z_i(v)$ ($Z_i(v) = \sum_{j\in A} v_j N_{ji}$) gives the mean number of times that the process is in state $i \in A$ before leaving region A and is therefore proportional to the steady distribution in A related to the absorbing process starting with the initial distribution v. Noting that $\tau_A(v) = \sum_{i\in A} Z_i(v)$ and $\sum_{i\in A, j\in B} Z_i(v)P_{ij} = 1$,[17] we easily recognize that

$$\frac{1}{\tau_A(v)} = \frac{\sum_{\substack{i\in A \\ j\in B}} Z_i(v)P_{ij}}{\sum_{i\in A} Z_i(v)} \equiv k_{A,B}(v)$$

where the last equivalence is established noting that the mathematical form of the central term is equivalent to the Kramers' rate constant, defined as the net flux out of A normalized by the population inside A.[4,35] A more detailed and general proof of the equivalence between mean escape time and Kramers' rate is presented in ref 35.

The assumption of a separation between the time scales of equilibration inside state A and escaping from it is formalized here assuming that the quantities $Z_i(v)$ do not depend on the starting distribution v and the ratio between any two of them is equal to the ratio between the steady-state probabilities π_i corresponding to the same nodes, in formula

$$\frac{Z_i}{Z_j} = \frac{\pi_i}{\pi_j} \qquad \forall\ i, j$$

From this assumption, we recognize that the ratio $\gamma = \pi_j/Z_j$ does not depend on node j and we have

$$k_{A,B} = \frac{\gamma \sum_{\substack{i\in A \\ j\in B}} Z_i P_{ij}}{\gamma \sum_{i\in A} Z_i} = \frac{\sum_{\substack{i\in A \\ j\in B}} \pi_i P_{ij}}{\sum_{i\in A} \pi_i} = \frac{Q_{A,B}}{\pi_A}$$

The escape time τ_A could be compared with the value $\pi_A/Q_{A,B}$, if the difference

$$\ln\left(\frac{Q_{A,B}}{\pi_A}\right) - \ln\left(\frac{1}{\tau_A}\right)$$

does not approach zero, then the assumption $Z_i/Z_j = \pi_i/\pi_j$ does not hold; hence, we discard the hypothesis of separated time scales. Note that the equality $-\ln(Q_{A,B}/\pi_A) = \ln \tau_A$ is necessary but not sufficient for the separated time scales condition: there are cases where the equality is true but the time scales are not separated, and cases where the equality holds only under a limit operation (see the Examples section).

EXAMPLES

A Degenerate State. Here, we present an example of the arguments introduced in the last section. We define an ergodic Markov chain with a set of three states; we then show how, depending on the transition probabilities, it is possible to face a situation in which the equality $-\ln(Q_{A,B}/\pi_A) = \ln \tau_A$ holds without regard for the separated time scales condition, or a situation such that the above equality holds only under a limit condition (the same condition with which we have the time scales separation).

Let define the set of states $V \equiv \{1,2,3\}$ and the transition matrix

$$P = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1-\delta-\omega & \delta & \omega \\ \delta & 1-\delta-\eta & \eta \\ \omega & \eta & 1-\omega-\eta \end{pmatrix} \end{array}$$

where $\delta > \omega > \eta$. The Markov chain described by P is irreducible and aperiodic; it is easy to see that the unique steady state is $\pi = {}^1/_3(1, 1, 1)$ and that the detailed balance condition

$\pi_i P_{ij} = \pi_j P_{ji}$ holds. The matrix of rate probabilities $Q_{ij} \equiv \pi_i P_{ij}$ is then $Q = {}^1/_3 P$.

Because of the small size of the problem, the minimum cut matrix $Q^\dagger$ is easily calculated by examining the values of all possible cuts, which results in

$$Q^\dagger = \begin{pmatrix} 0 & \delta+\eta & \eta+\omega \\ \delta+\eta & 0 & \eta+\omega \\ \eta+\omega & \eta+\omega & 0 \end{pmatrix}$$

From $Q^\dagger$, the free energy of the TS between any couple of nodes is defined as $G^\dagger = -\ln Q^\dagger$ (where, here, $-\ln 0$ is defined as 0). From the dendrogram associated with $G^\dagger$ (see Figure 2, top), it is possible to see the three possible ways to subdivide the chain in states according to the kinetics of the system. Here, we consider the partition scheme $\{\{1,2\},\{3\}\}$ and we define the cut (A,B), where $A \equiv \{1,2\}$ and $B \equiv \{3\}$. The cut value is $Q_{A,B} = {}^1/_3(\omega+\eta)$; we then have

$$\frac{Q_{A,B}}{\pi_A} = \frac{\omega+\eta}{2}$$

The mean escape time τ_A from region A is defined as $\tau_A \equiv \nu N \varepsilon$, where ν is the probability distribution of the starting nodes (here, we choose $\nu \equiv {}^1/_2(1, 1)$ for nodes in A), ε is the column vector with all entries 1, and N is the fundamental matrix of the absorbing Markov chain:

$$N \equiv (\mathrm{II} - P_A)^{-1} = \frac{1}{(\delta+\omega)(\delta+\eta) - \delta^2}\begin{pmatrix} \delta+\eta & \delta \\ \delta & \delta+\omega \end{pmatrix}$$

The resulting mean escape time is $\tau_A = (4\delta + \eta + \omega)/(2\delta\eta + 2\delta\omega + 2\eta\omega)$.

Some observations are needed. The probabilities ω and η are responsible for the transitions between A and B, so different values of them cause different scenarios. For example, in the case $\omega = \eta$, we obtain $Q_{A,B}/\pi_A = {\tau_A}^{-1} = \omega$, regardless of the value of δ, which means that there could be no separation between the time to equilibrate inside A and the time to escape from it. As already mentioned, the equality $Q_{A,B}/\pi_A = {\tau_A}^{-1}$ is not sufficient for the condition of separated time scales.

There are also cases where the above equality holds only under a limit operation (for example, in the case of $\eta = 0$ and $\omega \to 0$). In this situation ($\eta = 0$, $\omega > 0$, as depicted in the lower part of Figure 2), we have

$$-\ln\left(\frac{Q_{A,B}}{\pi_A}\right) = -\ln\left(\frac{\omega}{2}\right)$$

$$-\ln\left(\frac{1}{\tau_A}\right) = \ln\left(\frac{4\delta+\omega}{2\delta\omega}\right)$$

$$\Delta \equiv \ln\left(\frac{Q_{A,B}}{\pi_A}\right) - \ln\left(\frac{1}{\tau_A}\right) = \ln\left(\frac{Q_{A,B}}{\pi_A}\tau_A\right) = \ln\left(1 + \frac{\omega}{4\delta}\right)$$

The above equalities show that the ratio between $Q_{A,B}/\pi_A$ and ${\tau_A}^{-1}$ goes to 1 for $\omega \to 0$; in the same limit, we obtain the time scales separation. It is easy to see that, with $\delta = 0.5$ and $\Delta < 0.05$, we have $\omega < 0.1$ and $-\ln Q_{A,B}/\pi_A > 2.97$ (in units of $k_B T$). Hence, within this system, which could be considered to be composed of a degenerate state (made of two states) and another state, the approximation $Q_{A,B}/\pi_A \simeq {\tau_A}^{-1}$ (the difference is Δ) holds for a barrier $\Delta G_{A,B} > 3k_B T$. Under such conditions ($\delta = 0.5$ and $\omega = 0.1$), the chain could be reduced in the two-state system described by the master equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\pi_A(t) = \frac{\omega}{2}\pi_A(t) - \omega\pi_B(t)$$

and is governed by the exponential distribution $P_{A,B}(t) = (1 - \exp(-\gamma t/\tau_A)$, denoting the probability to see a jump from A to B in the interval of time $(0,t)$. Note that γ has units of inverse of time and it defines the time scale between the number of steps in the Markov chain and the corresponding time t for the Markov process: $n = \gamma t$.

A Complex MSM. Here, we apply the cut-based analysis on the MSM describing the reversible folding of the 20-residue three-stranded antiparallel β-sheet peptide studied in ref 22. The molecular dynamics sampling has been clustered according to backbone dihedral angles values, by means of a hierarchical algorithm[36] implemented in the molecular modeling package CAMPARI.[37] The resulting network has $V = 157\,380$ nodes and $E = 329\,011$ edges; it has been analyzed with PYKOV (a Markov chain Python module).[38] Because of the size of the network, instead of calculating the V^2 mincuts (which is a problem of complexity $O(V^4)$),[39] we selected the reference state by means of the cbFEP method[16] with ordering of the nodes according to mean first passage time to the folded state. This procedure is motivated by the fact that the cbFEP offers an approximated solution for the problem to collect nodes in states, it is an approximation because there is no guarantee to find the minimum cut and so the free energy of the TS calculated by cbFEP is lower or equal to the free energy of the TS derived from the minimum cut ($Q^\dagger_{A,B|i,j}$). At the cut (A,B), located at $\pi_A \simeq 0.33$ and separating the reference state A from all the rest B (see top of Figure 3), the TS and the activation free energies have values (in units of $k_B T$) of $G_{A,B} \simeq 5.69$ and $\Delta G_{A,B} = G_{A,B} - G_A \simeq 4.57$, respectively. Let ν be the restriction of π on the state A,

$$\nu_i = \frac{\pi_i}{\pi_A} \qquad \forall\, i \in A$$

then the mean escape time from state A, calculated by

$$\tau_A(\nu) = \sum_{i\in A} \nu_i \sum_{j\in A} N_{ij}\varepsilon_j$$

is $\tau_A \simeq 104$ steps (around 2 ns as the saving frequency was 20 ps), and its logarithm ($\ln \tau_A \simeq 4.64$) is similar to the activation free energy $\Delta G_{A,B}$, since the difference is much smaller than their absolute values. Moreover the distance $|Z(\nu)/\tau_A - \nu|$ (see below) is <0.02, where the vector $Z_i(\nu) = \sum_{j\in A}\nu_j N_{ji}$ indicates the mean number of times that the system is in node i before escaping from A. These results suggest that the system spends much less time to equilibrate inside state A than the time needed to escape from it. This observation can be further validated by means of the mixing time of state A, which is calculated by the following procedure. The subnetwork related to state A is extracted from the entire network. Since this subnetwork is not ergodic we then used its largest strongly connected component[40] as the network representing state A, which covers the 99% of the extracted network and, upon normalization, defines the MSM $\tilde{P}$ related to state A. We then

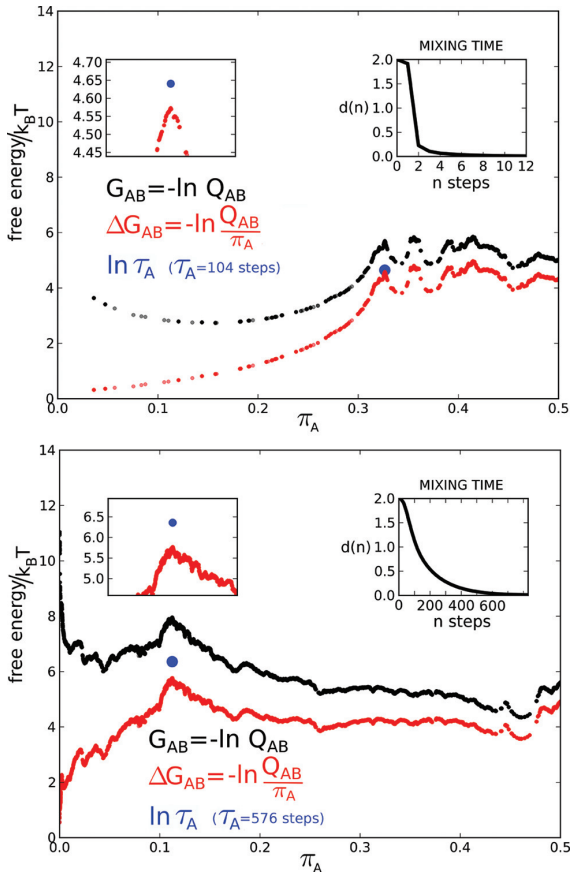


Figure 3. Application to a MSM of peptide folding. (Top) The folded state of a 20-residue β-sheet peptide was determined by the cbFEP.[22] The logarithm of the mean time to exit from the native state (blue dot) is essentially identical to the height of the activation free energy (red profile), suggesting that the system spends much less time to equilibrate inside the state than the time needed to escape from it. The time to mix within the native state (inset on the right) is much smaller than the mean exit time τ_A, as expected. (Bottom) Cut (A,B) of a non-native region of the phase space that has helical secondary structure content and is stabilized entropically.[22] The logarithm of the mean escape time from A does not overlap with the peak of the activation free energy, which implies that the system escapes before reaching equilibrium inside A. The mixing time (inset on the right) is not significantly shorter but rather similar to the mean escape time τ_A.

calculated the mixing time of $\tilde{P}$ in the following way: given the steady state $\tilde{\pi}$ of $\tilde{P}$, we define the initial v as the probability vector having value 1 at the least-probable node of $\tilde{\pi}$ and zeros for all the others. We then iterate vector v, $v(n) \equiv v\tilde{P}^n$, and for every n, we calculate the distance from the steady state:

$$d(n) \equiv |v(n) - \tilde{\pi}| \equiv \sum_i |v(n)_i - \tilde{\pi}_i|$$

($d(n)$ is weakly monotonically decreasing in n: $|v(n) - \tilde{\pi}| \geq |v(n+1) - \tilde{\pi}|$). As shown in Figure 3 top, $d(n)$ reaches zero in few steps, and defining the mixing time τ_{mix} as the smaller n such that $d(n) < 0.25$, we have that the time to mix inside state A is much smaller than the mean time to exit from it: $\tau_{mix} \simeq 2$ steps $\ll \tau_A$. Note that state A is the native state, but we refer to ref 22 for a description of the structures according to the position of the cut and for a detailed characterization of the free energy surface of folding of this β-sheet peptide.

In the bottom portion of Figure 3, the same analysis is performed on a different cut (A,B) concerning the helical state, a non-native region of the phase space, which is stabilized mainly by entropy.[22] In this case, the logarithm of the mean escape time from region A is not comparable with the activation free energy, so we do not expect a mixing time much smaller than the escape time. The two time scales are indeed of the same order. It also emerges that the distance $|Z(v)/\tau_A - v| \simeq 0.2$, which provides further evidence that the time to equilibrate within this non-native region is not negligible.

Cut-Based Free Energy Profile (cbFEP) of the Free Energy of Activation. The most significative information derived from the cbFEP analysis is contained in the peak coordinates of the first barrier. The x-value is the probability π_A of the state A delimited by the barrier, while the y-value represents either the free energy of the TS ($G_{A,B}$) or the free energy of activation ($\Delta G_{A,B}$) to exit from state A. In the previous example of a complex MSM, we showed how the cbFEP analysis of the free energy of activation ($\Delta G_{A,B}$), compared to the calculation of the mean exit time and mixing time of state A, is a useful tool to check for the existence of a metastable state. In fact, such comparison is able to evaluate the separation of time scales for equilibration within and exit from state A.

The cbFEP defined by $\Delta G_{A,B}$ is also advantageous, with respect to the profile of $G_{A,B}$, for another reason. While an exhaustive sampling of the phase space is required for the values of π_A and $G_{A,B}$ to be meaningful, this is not necessary for $\Delta G_{A,B}$. Indeed, being the free energy of activation the logarithm of the conditional probability $Q_{A,B}/\pi_A$, the sampling of the phase space far from the region of interest (i.e., far from the state A) is not required (see Figure 4). We assume here that the

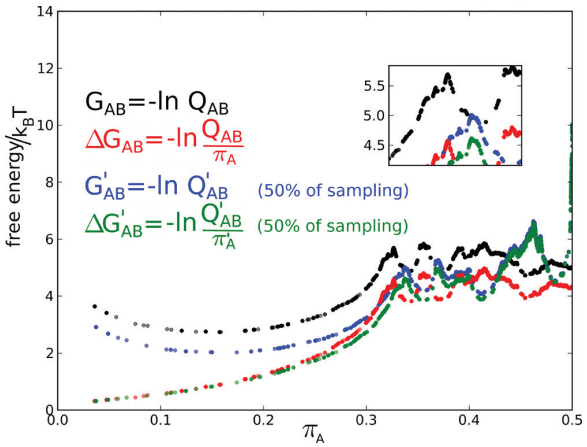


Figure 4. The free energy of activation is not affected by an incomplete sampling of the phase space. All the nodes located before $\pi_A = 0.5$ in the red (or black) profile define a subnetwork with half of the sampling of the original MSM. The subnetwork contains the native state and its barrier, but it does not contain the non-native region considered in the bottom of Figure 3. The quantity $Q'_{A,B}$ is the flow through the cut (A,B) calculated on the subnetwork (blue and green profiles). The figure shows that, at the peak of the first and most relevant barrier (inset on the figure), $G'_{A,B}$ is significantly different from $G_{A,B}$, whereas $\Delta G'_{A,B}$ is similar to $\Delta G_{A,B}$. The x-axis range of $G'_{A,B}$ and $\Delta G'_{A,B}$ is rescaled by a factor of 0.5, to better compare the cbFEPs.

lack of a complete sampling entails that the calculated probabilities π_A and $Q_{A,B}$ can be approximated by the correct ones by a rescaling factor α, i.e., we are assuming the equalities $\pi_A = \alpha\pi'_A$ and $Q_{A,B} = \alpha Q'_{A,B}$. This assumption is justified by the fact that the maximum likelihood probability $Q_{A,B}$, as well as π_A, is defined as the ratio $N_{A,B}/N$ between the number $N_{A,B}$ of observed transitions through the cut (A,B) and the total number N of transitions.[40] Whereas N is affected by a lack of sampling of noninteresting phase space regions, because the exploration of a restricted region needs less sampling, the amount $N_{A,B}$ of observed transitions through the cut must remain constant. In formulas, we have that $Q_{A,B} \equiv N_{A,B}/N = N_{A,B}/(N' - k)$, where N' indicates the total number of observed transitions for a sampling of a larger phase space region and k is the difference $N' - N$. The above assumption is now easily recovered, noting that $N_{A,B}/(N' - k) = \alpha Q'_{A,B}$, with $\alpha = N'/(N' - k)$. This heuristic reasoning suggests that, while $G_{A,B}$ and π_A are each individually affected by the lack of sampling, under the above assumption, $\Delta G_{A,B}$ is not influenced, because the α factors at the numerator and denominator of the $Q_{A,B}/\pi_A$ quotient cancel out.

CONCLUSION

Markov state models (MSMs) offer a relatively easy and powerful mathematical framework within which to define and analyze the kinetics of a complex system. The cut-based analysis of a MSM is based on the evaluation of the minimum cut between two nodes i and j, calculated as the cut (A,B) separating node $i \in A$ from node $j \in B$ with minimum flow through it. The cut-based analysis has, as the main objective, the study of the free energy surface to derive kinetic observables. We have shown here that the cut-based free energy of the transition state (TS) defines an ultrametric distance on the set of nodes of an ergodic MSM, without the assumption of the detailed balance condition. This property offers a way to collect nodes in states, which is a simplification procedure motivated by the kinetics of the system. The time scale to equilibrate inside a state is expected to be much smaller than the time to escape from it, and such difference can be checked by analytical calculations on the MSM. Kinetic observables like the free energy of the TS, the activation free energy, and the rate constants are directly derived from the cut-based analysis. In the same direction, here, we have proposed a novel definition of the cut-based free energy profile (cbFEP) that allows one to check for the separation of time scales for equilibration within and exit from a state. In the framework of protein dynamics, the final target of such analysis is to compare simulation results with the corresponding experimental observables, and, in this sense, it was our intention to make explicit the existence of a parallelism between cut-based quantities (TS and activation free energies) and the concepts at the base of transition state theory (TST) and Kramers' theory.

Lastly, it is essential to mention the wide applicability of the ultrametric distance defined here; given that it is defined on the set of nodes of an ergodic Markov chain, its application is not dependent on the physical system under study and could prove to be useful in fields far from protein folding, e.g., material sciences and bioinformatics.

AUTHOR INFORMATION

Corresponding Author

*E-mail: caflisch@bioc.uzh.ch.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

We thank Dr. Andreas Vitalis for interesting discussions. This work was supported by a grant of the Swiss National Science Foundation to author A.C.

REFERENCES

(1) Zwanzig, R. *Proc. Natl. Acad. Sci., U.S.A.* **1997**, *94*, 148−150.
(2) Zwanzig, R. *Proc. Natl. Acad. Sci., U.S.A.* **1995**, *92*, 9801−9804.
(3) Zwanzig, R. *Nonequilibrium Statistical Mechanics*; Oxford University Press: New York: 2001.
(4) Hänggi, P.; Talkner, P.; Borkovec, M. *Rev. Mod. Phys.* **1990**, *62*, 251−341.
(5) Kramers, H. A. *Physica* **1940**, *7*, 284.
(6) Pande, V. S. *Phys. Rev. Lett.* **2010**, *105*, 198101.
(7) Beauchamp, K. A.; Bowman, G. R.; Lane, T. J.; Maibaum, L.; Haque, I. S.; Pande, V. S. *J. Chem. Theory Comput.* **2011**, *7*, 3412−3419.
(8) Chodera, J. D.; Singhal, N.; Pande, V. S.; Dill, K. A.; Swope, W. C. *J. Chem. Phys.* **2007**, *126*, 155101.
(9) Keller, B.; Hünenberger, P.; van Gunsteren, W. F. *J. Chem. Theory Comput.* **2011**, *7*, 1032−1044.
(10) Buchete, N.-V.; Hummer, G. *Phys. Rev. E* **2008**, *77*, 030902.
(11) Noé, F.; Fischer, S. *Curr. Opin. Struct. Biol.* **2008**, *18*, 154−162.
(12) Pande, V. S.; Beauchamp, K.; Bowman, G. R. *Methods* **2010**, *52*, 99−105.
(13) Krivov, S. V.; Karplus, M. *Proc. Natl. Acad. Sci., U.S.A.* **2004**, *101*, 14766−14770.
(14) McGarrity, E. S.; Duxbury, P. M.; Holm, E. A. *Phys. Rev. E* **2005**, *71*, 026102.
(15) Gfeller, D.; De Los Rios, P.; Caflisch, A.; Rao, F. *Proc. Natl. Acad. Sci., U.S.A.* **2007**, *104*, 1817−1822.
(16) Krivov, S. V.; Karplus, M. *J. Phys. Chem. B* **2006**, *110*, 12689−12698.
(17) Kemeny, J. G.; Snell, J. L. *Finite Markov Chains*; Springer−Verlag: Berlin, Germany, 1976.
(18) Jungnickel, D. *Graphs, Networks and Algorithms*; Springer−Verlag: Berlin, Germany, 2005.
(19) Ford, L.; Fulkerson, D. R. *Can. J. Math.* **1956**, *8*, 399−404.
(20) Prinz, J.-H.; Wu, H.; Sarich, M.; Keller, B.; Senne, M.; Held, M.; Chodera, J. D.; Schütte, C.; Noé, F. *J. Chem. Phys.* **2011**, *134*, 174105.
(21) Rammal, R.; Toulouse, G.; Virasoro, M. A. *Rev. Mod. Phys.* **1986**, *58*, 765−788.
(22) Krivov, S. V.; Muff, S.; Caflisch, A.; Karplus, M. *J. Phys. Chem. B* **2008**, *112*, 8701−8714.
(23) Schuetz, P.; Wuttke, R.; Schuler, B.; Caflisch, A. *J. Phys. Chem. B* **2010**, *114*, 15227−15235.
(24) Becker, O. M.; Karplus, M. *J. Chem. Phys.* **1997**, *106*, 1495−1517.
(25) Wales, D.; Miller, M.; Walsh, T. *Nature* **1998**, *394*, 758−760.
(26) Krivov, S. V.; Karplus, M. *J. Chem. Phys.* **2002**, *117*, 10894−10903.
(27) Rylance, G. J.; Johnston, R. L.; Matsunaga, Y.; Li, C.-B.; Baba, A.; Komatsuzaki, T. *Proc. Natl. Acad. Sci., U.S.A.* **2006**, *103*, 18551−18555.
(28) Gomory, R.; Hu, T. *J. Soc. Ind. Appl. Math.* **1961**, *9*, 551−570.
(29) Liggett, T. M. *Continuous Time Markov Processes: An Introduction*; Cox, D., Krantz, S., Mazzeo, R., Scharlemann, M., Eds.; Graduate Studies in Mathematics, Vol. 113; American Mathematical Society: Providence, RI, 2010.
(30) Rozanov, Y. *Introduction to Random Processes*; Springer−Verlag: Berlin, Germany, 1982.
(31) Bovier, A.; Eckhoff, M.; Gayrard, V.; Klein, M. *Commun. Math. Phys.* **2002**, *228*, 219−255.
(32) Larralde, H.; Leyvraz, F. *Phys. Rev. Lett.* **2005**, *94*, 160201.

(33) Avetisov, V.; Bikulov, A. *Tr. Math. Inst. Steklova* **2009**, *265*, 82−89.
(34) Beltrán, J.; Landim, C. *Stoch. Proc. Appl.* **2011**, *121*, 1633−1677.
(35) Reimann, P.; Schmid, G. J.; Hänggi, P. *Phys. Rev. E: Stat. Phys., Plasmas, Fluids, Relat. Interdiscip. Top.* **1999**, *60*, R1−R4.
(36) Vitalis, A.; Caflisch, A. *J. Chem. Theory Comput.* **2012**, *8*, 1108−1120 (DOI:10.1021/ct200801b).
(37) Vitalis, A.; Pappu, R. *Ann. Rep. Comput. Chem.* **2009**, *5*, 49−76.
(38) Scalco, R. page http://riccardoscalco.github.com/Pykov/ (accessed February 20, 2012).
(39) Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C. *Introduction to Algorithms*; MIT Press: Cambridge, MA, 2001.
(40) Scalco, R.; Caflisch, A. *J. Phys. Chem. B* **2011**, *115*, 6358−6365.

Chapter 4

Simplified sequences of Protein G

Most globular proteins fold from a broad ensemble of denatured conformations to a unique three-dimensional structure which is the spatial arrangement of backbone and side chain atoms required to be functional. Even for small and single-domain proteins the folding process is complex because of the many degrees of freedom and the importance of both enthalpic and entropic contributions to the free energy [1]. Proteins in "modern" organisms are the result of evolution, which suggests to investigate the natural selection of amino acid alphabets and sequences to shed light on the complexity of the folding process. Evolution has selected sequences for specific biological functions, which, except for the natively unfolded proteins, require a thermodynamically stable folded structure [2]. Although the folding speed is not under direct evolutionary pressure, fast folding (i.e., in the microsecond to second time scale) is necessary for many biological functions that have to be fine-tuned in time, such as signal transduction and rapid adaptation to changes in the environment. Concerning the requirement of a stable folded structure it has been suggested that a sufficiently high diversity of interactions is required for folding to a unique state with an energy much more favorable than "decoy" structures [3, 4]. Diversity of interactions requires an heterogeneous amino acid alphabet. Theoretical analysis and compute simulations have suggested that selection of sequences that yield a native conformation with a pronounced energy minimum, i.e., an energy gap with respect to other structures, solves the problem of kinetic accessibility of the native conformation [5, 6].

Previously, we had investigated a highly simplified variant of the sequence of the immunoglobulin-binding domain of protein G (a 56-residue α/β fold) by replacing it with polyalanine, polythreonine, and diglycine segments at regions of the sequence that in the folded structure are α-helical, β-strand, and turns, respectively. Remarkably, multiple folding and unfolding events were observed in a 15-μs molecular dynamics simulation at 330 K. The most stable state (populated at about 20%) of the highly simplified-sequence variant of protein G has the same α/β topology as the wild type but shows the characteristics of a molten globule, i.e., loose contacts among side chains and lack of a specific hydrophobic core. The unfolded state is heterogeneous and includes a variety of α/β topologies but also fully α-helical and fully β-sheet structures. Transitions within the denatured state are very fast, and the molten-globule state is reached in less than 1 μs by a framework mechanism of folding with multiple pathways [7]. Here we use atomistic simulations to investigate the folded state and (un)folding

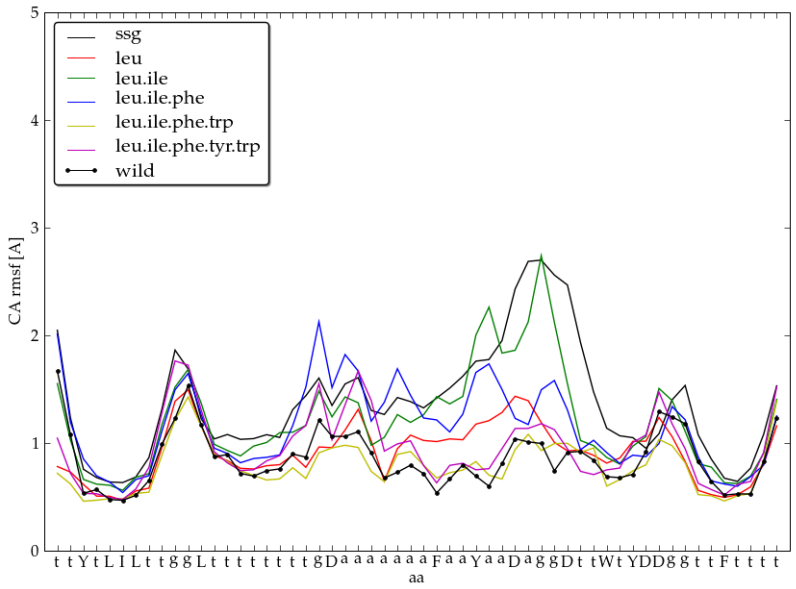


Figure 4.1: The flexibility of the folded state anticorrelates with the complexity of the sequence: higher the complexity of the sequence the more rigid is the native structure.

mechanism of a series of variants of protein G with increasing amount of sequence complexity. The present study on primordial proteins was motivated by the following questions: How does the free energy surface of the native basin change by increasing the complexity of the sequence? Does the folding process shift from a diffusion-collision mechanism (due to the strong secondary structure propensity of highly simplified sequences) to a nucleation-condensation model? Did the denatured state of primordial proteins include conformations with native and/or non-native secondary structure elements and topologies?

A marked reduction in flexibility of the native structure is observed by increasing the number of residue types (i.e., sequence complexity) with both implicit solvent models as well as the explicit water simulations. The plots of Cα RMSF of the native structure show clearly that the higher the complexity of the sequence the more rigid is the native structure (figure 4.1). This trend is observed for the elements of regular secondary structure as well as the four loops. Moreover, both implicit solvent models and the explicit water simulations show essentially identical RMSF plots.

4.1 Analysis

Protein G: pdb file: **1PGB.pdb**, sequence:

MTYKLILNGKTLKGETTTEAVDAATAEKVFKQYANDNGVDGEWTYDDATKTFTVTE

Mutant ssG:

TTTTTTTTTGGTTTTTTTTTGGAAAAAAAAAAAAAAGGTTTTTTTTGGTTTTTTT

Other mutants: Inclusion of *all* native residues of *one or more types* to ssG. Asp is always included.

aa	positions
Asp	22,36,40,46,47
Leu	5,7,12
Ile	6
Phe	30,52
Tyr	3,33,45
Trp	43

Example: **ile.leu**

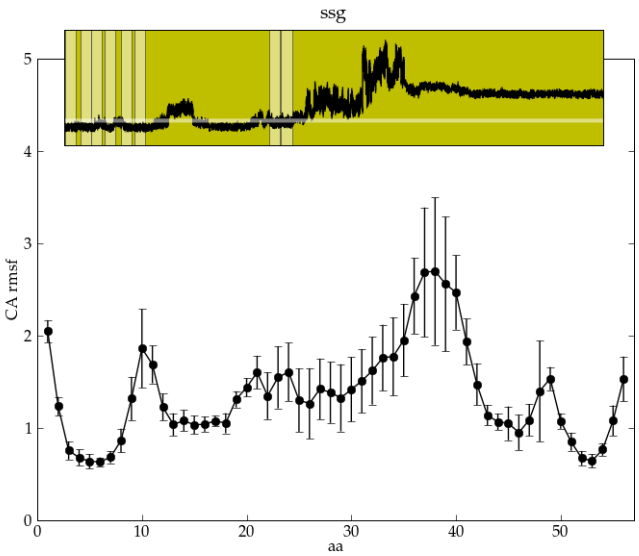
TTTT**LIL**TTGG**L**TTTTTTTTG**D**AAAAAAAAAAAAA**D**AGG**D**TTTTT**DD**GGTTTTTTT

On the following are presented the two analysis supporting the hypothesis that the flexibility of the native structure decreases by increasing the sequence complexity. The two analysis are: 1) the root mean square fluctuations (RMSF) of the residues around the native position and 2) the quantification of the native contacts and secondary structure formed during the simulation.

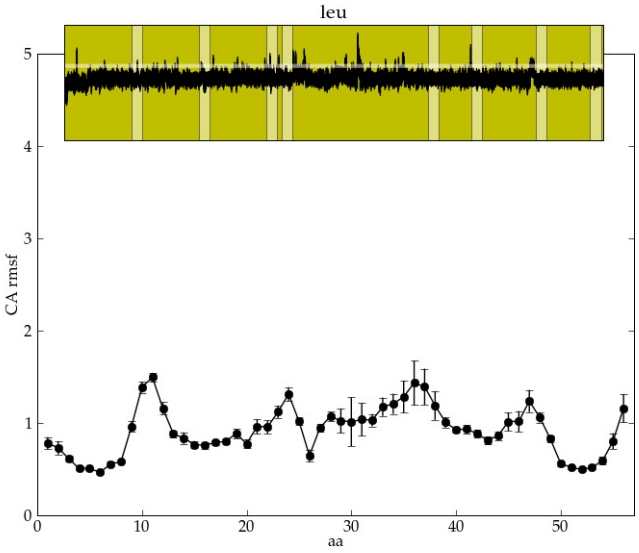
4.1.1 RMSF

The following plots show the rmsf values (in Å) of the native state. The plot on the top is the rmsd time series from the native structure and it indicates that the rmsf values are calculated in 8 windows of 1 ns (500 frames). In every window the rmsd average from the native is lower than 4 Å.

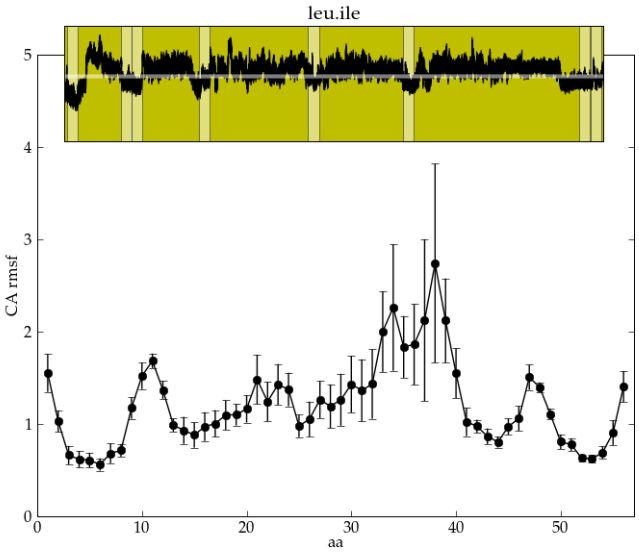
ssg


ssg
CA rmsf
aa
0
1
2
3
4
5
0
10
20
30
40
50

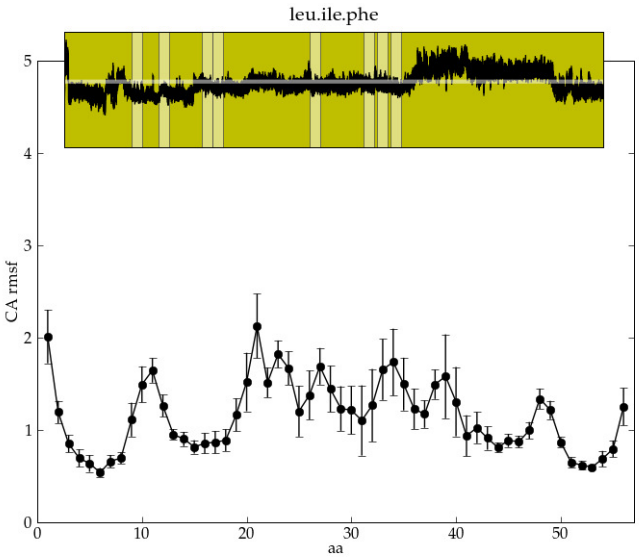
leu


leu
CA rmsf
aa
0
1
2
3
4
5
0
10
20
30
40
50

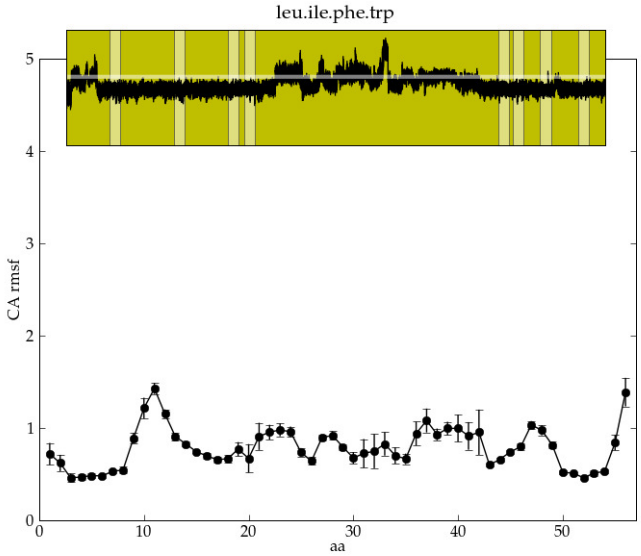
leu.ile



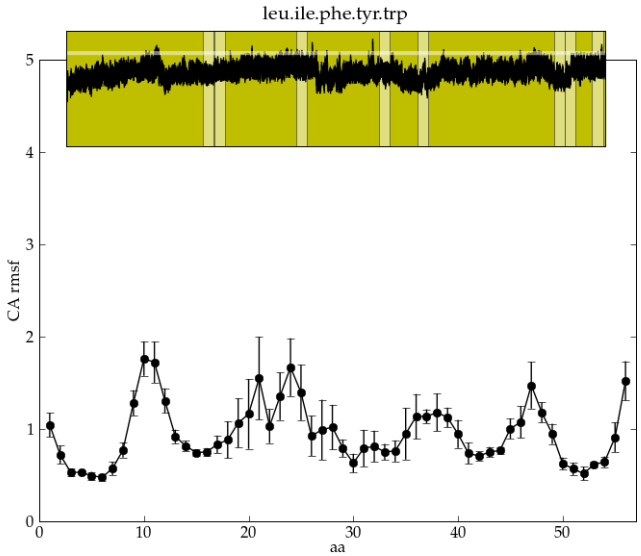
leu.ile.phe



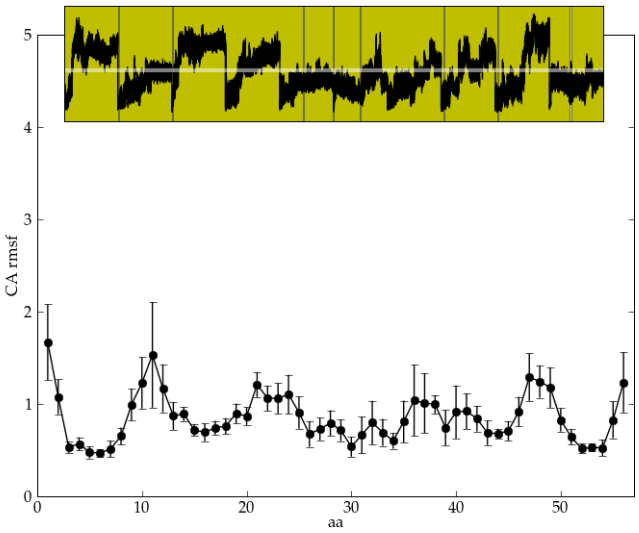
leu.ile.phe.trp



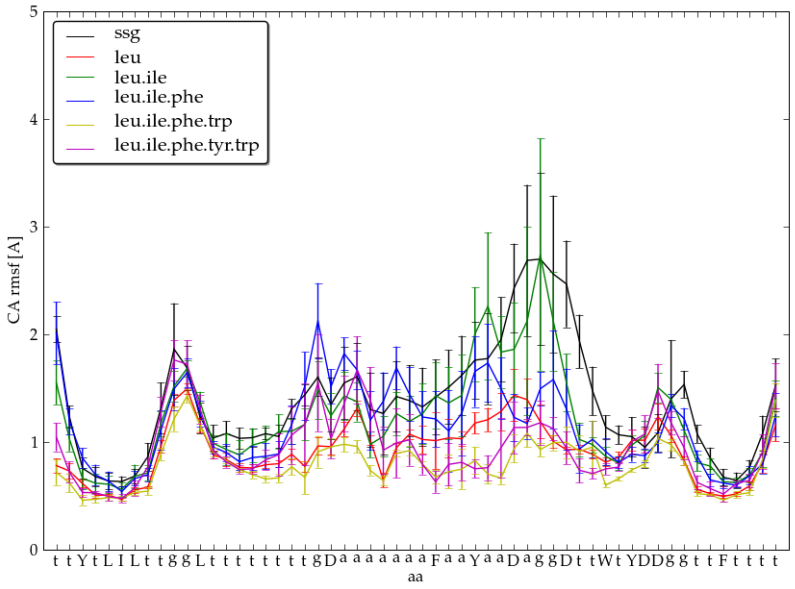
leu.ile.phe.tyr.trp



wild



enriching plot



The last plot shows the superposition of the rmsf values for the different mutants. The flexibility of the folded state anticorrelates with the complexity of the sequence: higher the complexity of the sequence the more rigid is the native structure.

4.1.2 Native contacts

The comparison between the amount of native contants and the secondary structure formed could reveal the mechanism of folding. Despite the following calculation is not a kinetic analysis, and therefore cannot give conclusive statements on the folding process, it is able to guide the ideas on how much the sequence complexity is relevant for what concerns a possible shift from a diffusion-collision mechanism (due to the strong secondary structure propensity of highly simplified sequences) to a nucleation-condensation model for the more complex sequences. It results that the ratio between the fraction of secondary structure formed divided by the fraction of native contacts goes to one by increasing the sequence complexity, which means that the secondary and tertiary structures are formed together (figure 4.2).

The analysis consists on the following evaluations:

- Q: fraction of native contacts on the mutant respect the wild protein. Number of native contacts formed divided by the number of native contacts expected.
- r^2: fraction of secondary structure formed respect native. The native seconday structure is, by default,

  'XEEEEEEEEXXEEEEEEEEXXXHHHHHHHHHHHHHHHHXXXXEEEEEXXXXEEEEEX'

  We look at $56 - 15$ aa along the sequence. The rate is the number of such 41 secondary elements formed divided by 41.
- $\frac{r^2}{Q}$: ratios between the fraction of secondary structure formed divided by the fraction of native contacts. Parameter *m* (in the following plot) means that only frames with both r^2 and Q greater than m are taken into account.

Note that it is considered only a snapshot every 100. Given the parameter NSAVC=10000 during similations, is more precise to say that only a snapshot every $100 * 10000 = 10^6$ is taken into account.

Given the ratios $\frac{r^2}{Q}$ for every frames, the set of all frames is dividev in 10 blocks and the means are then calculated. On the following there are the plots with the mean and error bars for every mutant:

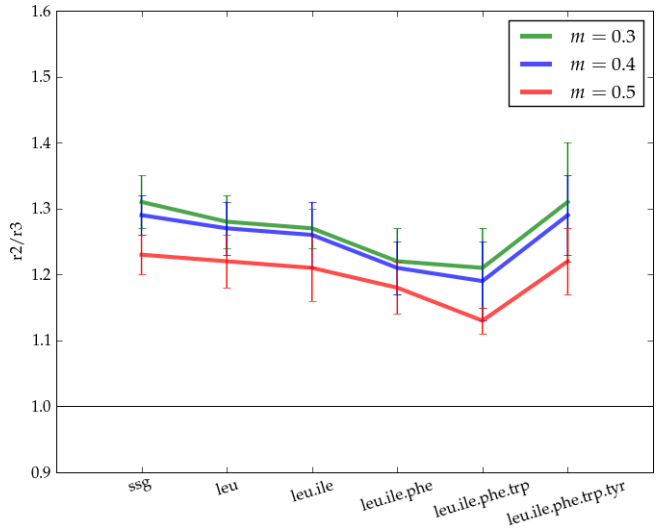


Figure 4.2: The plot shows the ratio between the fraction of secondary structure divided by the fraction of native contacts. Enriching the alphabet the ratio goes to one, which means that the secondary and tertiary structures tend to be formed together by increasing the sequence complexity.

Bibliography

[1] Karplus, M. (2000) Aspects of protein reaction dynamics: deviations from simple behavior. *J. Phys. Chem. B 104:1–27*

[2] Anfinsen, C. B. (1973) Principles that govern the folding of protein chains. *Science 181: 223–230*

[3] Shakhnovich, E. I. (1998) Protein design: a perspective from simple tractable models. *Folding & Design 3:R4–R58.*

[4] Shakhnovich, E. (2006) Protein folding thermodynamics and dynamics: where physics, chemistry, and biology meet. *Chem. Rev. 106:1559–1588*

[5] Shakhnovich, E. (1994) Proteins with selected sequences fold into unique native conformation. *Phys. Rev. Lett. 72:3907–3910*

[6] Sali, A. Shakhnovich, E. and Karplus, M. (1994) How does a protein fold? *Nature 369:248–251*

[7] Guarnera, E. Pellarin, R. and Caflisch, A. (2009) How does a simplified-sequence protein fold? *Biophys J. 97:173–46*

Conclusions and Outlook

Markov state models offer a powerful mathematical framework within which to define and analyze the kinetics of complex systems, like proteins. The contributions introduced with the present PhD thesis, in the part of computational biochemistry that uses the markovian language, may be summed up in the following points:

- The sampling of protein conformational space by multiple independent MD runs, possible in distributed computing, is usually biased because of the short length of each run and/or the choice of the starting conformation(s). The statistical bias can be formulated by an analytic expression that describes the dependence on the length of the trajectories, the choice of the starting conformation(s), and the underlying free energy surface.
- The unbiased distribution corresponds to the steady state distribution of the chain, which can be calculated if the Markov chain is ergodic. An automatic procedure is introduced for remove the minimum amount of links in order to have an ergodic chain from one that is not ergodic. The suggested algorithm was formulated by Tarjan and it is able to subdivide the network in its strongly connected components.
- The cut-based analysis of a MSM is based on the evaluation of the minimum cut between two nodes i and j, calculated as the cut (A, B) separating node $i \in A$ from node $j \in B$ with the minimum flow $Q_{A,B}$ through it. The cut-based analysis has the objective to study of the free energy surface to derive kinetic observables. It is possible to prove that the free energy of the transition state between nodes i and j defines an ultrametric distance on the set of nodes of an ergodic MSM, without the assumption of detailed balance condition.
- Both in transition state theory and in Kramers' theory the time scale to equilibrate inside a state is assumed to be much smaller than the time to escape from it. A novel definition of the cut-based free energy profile allows one to check for such separation of time scales by means of the comparison between the activation free energy to escape from a state and the mean time to exit from it.

Kinetics observables, like the free energy of the transition state, the activation free energy and the rate constants, are directly derived from the cut-based analysis. The final target is to compare simulation results with the corresponding experimental observables. With the same spirit, it is important for the future development of Markov chain applications in protein systems to maintain a solid parallelism between the concepts belonging to Markov theory and the experimental language. Moreover, it will be fundamental to use and develop statistical inference

methods able to accurately estimate the statistical uncertainty involved on the calculation, in order to better use the predictive power of the theory.

Appendix: Pykov documentation

Table Of Contents

Previous topic

Welcome to pykov's documentation!

This Page

Show Source

About Pykov

Pykov is Python module about finite Markov chains. You can define a Markov chain from scratch or read it from a text file according specific format. Pykov is versatile, being it able to manipulate the chain, inserting and removing nodes, and to calculate various kind of quantities, like the steady state distribution, mean first passage times, random walks, absorbing times, and so on. Pykov is also really fast, being it based on Pysparse, a sparse matrix library for Python.

Pykov is licensed under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

Click here to download Pykov.

Pykov would like to be a collaborative project. Any help is welcome, from a bug notification to a more demanding collaboration. Please refer to my github webpage for contacts.

Installing Pykov

You can install Pykov in different ways.

Pykov consists in a single Python module, so all you have to do is to download `pykov.py` from here and move it in your personal Python library.

Another solution is to unwload `pykov-1.0.tar.gz` (or `.zip`) from the pypi pykov page , unpack it, and—from the directory—run

```
python setup.py install
```

which will ultimately copy `pykov.py` to the appropriate directory in your Python installation.

Lastly, you can use *easy_install* to install it, simply run:

```
easy_install pykov
```

(please refer to the easy_install documentation to install and use *easy_install*.)

Dependences

Pykov depends on

- Python 2.7.1
- Numpy 1.5.1
- Pysparse 1.1

Warning Although a lower version for Python and Numpy could be fine, version 1.1 of Pysparse is strictly required.

Getting started

You can define a Markov chain from scratch or read it from a text file according specific format.

From scratch

It is easily possible to define a **`Chain`** instance considering that it inherits from python class **`dict`**.

```
>>> P = pykov.Chain()
>>> P[('A','B')]=.3
>>> P
{('A', 'B'): 0.3}
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T
{('B', 'A'): 1.0, ('A', 'B'): 0.3, ('A', 'A'): 0.7}
```

Reading a txt file

Here there are two possibilities, depending on the information contained on the txt file.

From a transition matrix

In the easiest case the file `/mypath/mat` contains the transition matrix, and it has the following format:

```
A A .7
A B .3
B A 1
```

Then, the corresponding **`Chain`** instance is created with the command:

```
>>> P = pykov.readmat('/mypath/mat')
>>> P
{('B', 'A'): 1.0, ('A', 'B'): 0.3, ('A', 'A'): 0.7}
```

From a finite chain

In the case the **`Chain`** instance must be created from a finite chain of states, the transition matrix is not fully defined. The following function defines the transition probabilities as the maximum likelihood probabilities calculated along the chain. Having the file `/mypath/trj` with the following format:

```
1
1
1
2
1
3
```

the **`Chain`** instance defined from that chain is:

```
>>> t = pykov.readtrj('/mypath/trj')
>>> t
(1, 1, 1, 2, 1, 3)
>>> p, P = maximum_likelihood_probabilities(t,lag_time=1, separator='0')
>>> p
{1: 0.6666666666666666, 2: 0.1666666666666666, 3: 0.1666666666666666}
>>> P
{(1, 2): 0.25, (1, 3): 0.25, (1, 1): 0.5, (2, 1): 1.0, (3, 3): 1.0}
>>> type(P)
<class 'pykov.Chain'>
>>> type(p)
<class 'pykov.Vector'>
```

Note Here there are the mathematical details. Let the chain be `(1,1,1,2,1,3)`, then the list of its links, with `lag-time=1`, is `[(1,1),(1,1),(1,2),(2,1),(1,3)]` (with `lag-time=2` the list is `[(1,1),(1,2),(1,1),(2,3)]`, which means that states separated by two steps are linked to each other). Before to defines the probabilities of interest, all the dead branchs are removed from the list of links, in the example the dead brach is in state `3` because `3` has no successors, in the sense that there are not links having state `3` as first state. The dead branch is removed inserting an autoloop, in the example the list of links becomes `[(1,1),(1,1),(1,2),(2,1),(1,3),(3,3)]`.

Let `q[(i,j)]` the probability to find the link `(i,j)` in the list, for example `q[(1,1)]` is:

$$q_{1,1} = \frac{2}{6}$$

Then the probability of state `i` and the transition probability between states `i` and `j` are the defined as:

$$p_i = \sum_j q_{ij}$$

$$P_{ij} = \frac{q_{ij}}{p_i}$$

The `separator` keyword has the following meaning: all the links that have `separator` as initial or final state are removed from the list.

Classes and Methods

Vector Class

The **`Vector`** class inherits from python class **`dict`**, which means it has the same behaviors and features of python dictionaries, with few exceptions. The states and the corresponding probabilities are the `keys` and the `values` of the dictionary, respectively.

Manipulation of a Vector

Definitions

```
>>> pykov.Vector({'A':.3, 'B':.7})
{'A':.3, 'B':.7}
>>> pykov.Vector(A=.3, B=.7)
{'A':.3, 'B':.7}
```

Getting and setting items

Warning States not belonging to the vector have zero probability. Do not use **`keys()`** method to eveluate the set of states of the Markov chain, use instead the **`states()`** method of **`Chain`** class.

```
>>> q = pykov.Vector(C=.4, B=.6)
>>> q['C']
0.4
>>> q['Z']
0.0
>>> 'Z' in q
False
```

Warning Setting a state to zero probability is like removing the state.

```
>>> q = pykov.Vector(C=.4, B=.6)
>>> q['Z']=.2
>>> q
{'C': 0.4, 'B': 0.6, 'Z': 0.2}
>>> q['Z']=0
>>> q
{'C': 0.4, 'B': 0.6}
```

Vector operations

Sum

A `Vector` instance can be added or substracted to another `Vector` instance.

```
>>> p = pykov.Vector(A=.3, B=.7)
>>> q = pykov.Vector(C=.5, B=.5)
>>> p + q
{'A': 0.3, 'C': 0.5, 'B': 1.2}
>>> p - q
{'A': 0.3, 'C': -0.5, 'B': 0.2}
>>> q - p
{'A': -0.3, 'C': 0.5, 'B': -0.2}
```

Product

A `Vector` instance can be multiplied by a `scalar`, another `Vector` and a `Chain` or `Matrix` instance.

```
>>> p = pykov.Vector(A=.3, B=.7)
>>> p * 3
{'A': 0.9, 'B': 2.1}
>>> 3 * p
```

```
{'A': 0.9, 'B': 2.1}
>>> q = pykov.Vector(C=.5, B=.5)
>>> p * q
0.35
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> p * T
{'A': 0.91, 'B': 0.09}
>>> T * p
{'A': 0.42, 'B': 0.3}
```

Methods

class pykov.**Vector**(*data=None*, ***kwargs*)

sum() [source]

Sum the values. [source]

```
>>> p = pykov.Vector(A=.3, B=.7)
>>> p.sum()
1.0
```

sort(*reverse=False*)

Sort according the probability. [source]

```
>>> p = pykov.Vector({'A':.3, 'B':.1, 'C':.6})
>>> p.sort()
[('B', 0.1), ('A', 0.3), ('C', 0.6)]
>>> p.sort(reverse=True)
[('C', 0.6), ('A', 0.3), ('B', 0.1)]
```

choose()

Choose a state according to its probability. [source]

```
>>> p = pykov.Vector(A=.3, B=.7)
>>> p.choose()
'B'
```

See also Kevin Parks recipe

normalize()

Normalize the vector so that the entries sum is 1. [source]

```
>>> p = pykov.Vector({'A':3, 'B':1, 'C':6})
>>> p.normalize()
>>> p
{'A': 0.3, 'C': 0.6, 'B': 0.1}
```

copy()

Return a shallow copy. [source]

```
>>> p = pykov.Vector(A=.3, B=.7)
>>> q = p.copy()
>>> p['C'] = 1.
>>> q
{'A': 0.3, 'B': 0.7}
```

entropy()

Return the entropy. [source]

$$H(p) = \sum_i p_i \ln p_i$$

See also Khinchin, A. I. Mathematical Foundations of Information Theory Dover, 1957.

```
>>> p = pykov.Vector(A=.3, B=.7)
>>> p.entropy()
0.6108643020548935
```

dist(*p*)

Return the distance between the two probability vectors. [source]

$$d(q,p) = \sum_i |q_i - p_i|$$

```
>>> p = pykov.Vector(A=.3, B=.7)
>>> q = pykov.Vector(C=.5, B=.5)
>>> q.dist(p)
1.0
```

relative_entropy(*p*)

Return the Kullback-Leibler distance. [source]

$$d(q,p) = \sum_i q_i \ln(q_i/p_i)$$

Note The Kullback-Leibler distance is not symmetric.

```
>>> p = pykov.Vector(A=.3, B=.7)
>>> q = pykov.Vector(A=.4, B=.6)
>>> p.relative_entropy(q)
0.021600854143546 54
>>> q.relative_entropy(p)
0.022582421084357485
```

Matrix Class

The **Matrix** class inherits from python class **dict**. The `dict keys` are `tuple` of indexes, the `dict values` are the matrix entries.

Note Indexes do not need to be `int`, they can be `string`.

Manipulation of a Matrix

Definitions

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
```

Getting and setting items

Note Couples of states with zero transition probability do not appear among the `keys`, but they get zero if asked. In other words, the `keys` set contains only the non-zero transitions.

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T[('A','B')]
0.3
>>> T['A','B']
0.3
>>> T['B','B']
0.0
```

```
>>> T = pykov.Matrix()
>>> T[('A','B')] = .3
>>> T
{('A', 'B'): 0.3}
>>> T['A','A'] = .7
>>> T
{('A', 'B'): 0.3, ('A', 'A'): 0.7}
>>> T['B','B'] = 0
>>> T
{('A', 'B'): 0.3, ('A', 'A'): 0.7}
>>> T['A','A'] = 0
>>> T
{('A', 'B'): 0.3}
```

Warning States without ingoing or outgoing transitions are removed from the set of states.

```
>>> T = pykov.Matrix({('A','B'): 3, ('A','A'): 7, ('B','A'): .1})
>>> T.states()
set(['A', 'B'])
>>> T['A','C']=1
>>> T.states()
set(['A', 'C', 'B'])
>>> T['A','C']=0
>>> T.states()
set(['A', 'B'])
```

Matrix operations

Sum

A `Matrix` instance can be added or substracted to another `Matrix` instance.

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> I = pykov.Matrix({('A','A'):1, ('B','B'):1})
>>> T + I
{('B', 'A'): 1.0, ('A', 'B'): 0.3, ('A', 'A'): 1.7, ('B', 'B'): 1.0}
```

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> I = pykov.Matrix({('A','A'):1, ('B','B'):1})
>>> T - I
{('B', 'A'): 1.0, ('A', 'B'): 0.3, ('A', 'A'): -0.3, ('B', 'B'): -1}
```

Product

A `Matrix` instance can be multiplied by a `scalar`, a `Vector` or another `Matrix` instance.

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T * 3
{('B', 'A'): 3.0, ('A', 'B'): 0.9, ('A', 'A'): 2.1}
>>> p = pykov.Vector(A=.3, B=.7)
>>> T * p
{'A': 0.42, 'B': 0.3}
>>> W = pykov.Matrix({('N', 'M'): 0.5, ('M', 'N'): 0.7,
                      ('M', 'M'): 0.3, ('O', 'N'): 0.5,
                      ('O', 'O'): 0.5, ('N', 'O'): 0.5})
>>> W * W
{('N', 'M'): 0.15, ('M', 'N'): 0.21, ('M', 'O'): 0.35,
 ('M', 'M'): 0.44, ('O', 'M'): 0.25, ('O', 'N'): 0.25,
 ('O', 'O'): 0.5, ('N', 'O'): 0.25, ('N', 'N'): 0.6}
```

Methods

class pykov.**Matrix**(*data=None*)

states() [source]

Return the set of states. [source]

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.states()
set(['A', 'B'])
```

pred(*key=None*)

Return the precedessors of a state (if not indicated, of all states). In Matrix notation: return the coloum of the indicated state. [source]

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.pred()
{'A': {'A': 0.7, 'B': 1.0}, 'B': {'A': 0.3}}
>>> T.pred('A')
{'A': 0.7, 'B': 1.0}
```

succ(*key=None*)

Return the successors of a state (if not indicated, of all states). In Matrix notation: return the row of the indicated state. [source]

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.succ()
{'A': {'A': 0.7, 'B': 0.3}, 'B': {'A': 1.0}}
>>> T.succ('A')
{'A': 0.7, 'B': 0.3}
```

copy()

Return a shallow copy. [source]

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> W = T.copy()
>>> T[('B','B')] = 1.
>>> W
{('B', 'A'): 1.0, ('A', 'B'): 0.3, ('A', 'A'): 0.7}
```

remove(*states*)

Return a copy of the Chain, without the indicated states. [source]

Warning All the links where the states appear are deleted, so that the result will not be in general a stochastic matrix.

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.remove(['B'])
{('A', 'A'): 0.7}
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.,
                     ('C','D'): .5, ('D','C'): 1., ('C','B'): .5})
>>> T.remove_from(['A','B'])
{('C', 'D'): 0.5, ('D', 'C'): 1.0}
```

stochastic()

Make a right stochastic matrix. [source]

Set the sum of every row equal to one, raise `PykovError` if it is not possible.

```
>>> T = pykov.Matrix({('A','B'): 3, ('A','A'): 7, ('B','A'): .2})
>>> T.stochastic()
>>> T
{('B', 'A'): 1.0, ('A', 'B'): 0.3, ('A', 'A'): 0.7}
>>> T[('A','C')]=1
>>> T.stochastic()
pykov.PykovError: 'Zero links from node C'
```

transpose()

Return the transpose Matrix. [source]

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.transpose()
{('B', 'A'): 0.3, ('A', 'B'): 1.0, ('A', 'A'): 0.7}
```

eye()

Return the Identity Matrix. [source]

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.eye()
{('A', 'A'): 1., ('B', 'B'): 1.}
```

ones()

Return a `Vector` instance with entries equal to one. [source]

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.ones()
{'A': 1.0, 'B': 1.0}
```

trace()

Return the Matrix trace. [source]

```
>>> T = pykov.Matrix({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.trace()
0.7
```

Chain Class

The **Chain** class inherits from the **Matrix** class. The `dict keys` are `tuple` of states, the `dict values` are the transition probability between the two states. The **Chain** class methods are

Methods

class `pykov`.**Chain**(*data=None*)

adiacence() [source]

Return the adiacence matrix. [source]

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.adiacence()
{('B', 'A'): 1, ('A', 'B'): 1, ('A', 'A'): 1}
```

pow(*p*, *n*)

Find the probability distribution after n steps, starting from an initial `Vector`. [source]

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> p = pykov.Vector(A=1)
>>> T.pow(p,3)
{'A': 0.7629999999999999, 'B': 0.23699999999999996}
>>> p * T * T * T
{'A': 0.7629999999999999, 'B': 0.23699999999999996}
```

move(*state*)

Do one step from the indicated state, and return the final state. [source]

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.move('A')
'B'
```

walk(*steps*, *start=None*, *stop=None*)

Return a random walk of n steps, starting and stopping at the indicated states. [source]

Note If not indicated, then the starting state is chosen according to its steady probability. If the stopping state is reached before to do n steps, then the walker stops.

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.walk(10)
['B', 'A', 'B', 'A', 'A', 'B', 'A', 'A', 'A', 'B', 'A']
>>> T.walk(10,'B','B')
['B', 'A', 'A', 'A', 'A', 'A', 'B']
```

walk_probability(*walk*)

Given a walk, return the log of its probability. [source]

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.walk_probability(['A','A','B','A','A'])
-1.917322692203401
>>> probability = math.exp(-1.917322692203401)
0.147
>>> p = T.walk_probability(['A','B','B','B','A'])
>>> math.exp(p)
0.0
```

steady()

With the assumption of ergodicity, return the steady state. [source]

Note Inverse iteration method (P is the Markov chain)

$$Q = \mathbf{I} - P$$

$$Q^T x = e$$
$$e = (0, 0, \ldots, 0, 1)$$

See also W. Stewart: Introduction to the Numerical Solution of Markov Chains, Princeton University Press, Chichester, West Sussex, 1994.

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.steady()
{'A': 0.7692307692307676, 'B': 0.23076923076923028}
```

mixing_time(*cutoff=0.25*, *jump=1*, *p=None*)

Return the mixing time. [source]

If the initial distribution (p) is not indicated, then it is set to p={'less probable state':1}.

Note The mixing time is calculated here as the number of steps (n) needed to have

$$|p(n) - \pi| < 0.25$$
$$p(n) = pP^n$$
$$\pi = \pi P$$

The parameter `jump` controls the iteration step, for example with `jump=2` n has values 2,4,6,8,..

```
>>> d = {('R','R'):1./2, ('R','N'):1./4, ('R','S'):1./4,
         ('N','R'):1./2, ('N','N'):0., ('N','S'):1./2,
         ('S','R'):1./4, ('S','N'):1./4, ('S','S'):1./2}
>>> T = pykov.Chain(d)
>>> T.mixing_time()
2
```

entropy(*p=None*, *norm=False*)

Return the `Chain` entropy, calculated with the indicated probability Vector (the steady state by default). [source]

$$H_i = \sum_j P_{ij} \ln P_{ij}$$
$$H = \sum \pi_i H_i$$

See also Khinchin, A. I. Mathematical Foundations of Information Theory Dover, 1957.

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.entropy()
0.46989561696530169
```

With normalization entropy belongs to [0,1]

```
>>> T.entropy(norm=True)
0.33895603665233132
```

mfpt_to(*state*)

Return the Mean First Passage Times of every state to the indicated state. [source]

See also Kemeny J. G.; Snell, J. L. Finite Markov Chains. Springer-Verlag: New York, 1976.

```
>>> d = {('R', 'N'): 0.25, ('R', 'S'): 0.25, ('S', 'R'): 0.25,
         ('R', 'R'): 0.5, ('N', 'S'): 0.5, ('S', 'S'): 0.5,
         ('S', 'N'): 0.25, ('N', 'R'): 0.5, ('N', 'N'): 0.0}
>>> T = pykov.Chain(d)
>>> T.mfpt_to('R')
{'S': 3.333333333333333, 'N': 2.666666666666667}
```

absorbing_time(*transient_set*)

Mean number of steps needed to leave the transient set. [source]

Return the `Vector tau`, the `tau[i]` is the mean number of steps needed to leave the transient set starting from state `i`. The parameter `transient_set` is a subset of nodes.

Note If the starting point is a `Vector p`, then it is sufficient to calculate `p * tau` in order to weigh the mean times according the initial conditions.

```
>>> d = {('R','R'):1./2, ('R','N'):1./4, ('R','S'):1./4,
         ('N','R'):1./2, ('N','N'):0., ('N','S'):1./2,
         ('S','R'):1./4, ('S','N'):1./4, ('S','S'):1./2}
>>> T = pykov.Chain(d)
>>> p = pykov.Vector({'N':.3, 'S':.7})
>>> tau = T.absorbing_time(p.keys())
>>> p * tau
3.1333333333333329
```

absorbing_tour(*p*, *transient_set=None*)

Return a `Vector v`, `v[i]` is the mean of the total number of times the process is in a given transient state `i` before to leave the transient set. [source]

Note `v.sum()` is equal to `p * tau` (see **absorbing_time()** method).

In not specified, the `transient set` (with its probability) is defined by means of the `Vector p`.

See also Kemeny J. G.; Snell, J. L. Finite Markov Chains. Springer-Verlag: New York, 1976.

```
>>> d = {('R','R'):1./2, ('R','N'):1./4, ('R','S'):1./4,
         ('N','R'):1./2, ('N','N'):0., ('N','S'):1./2,
         ('S','R'):1./4, ('S','N'):1./4, ('S','S'):1./2}
>>> T = pykov.Chain(d)
>>> p = pykov.Vector({'N':.3, 'S':.7})
>>> T.absorbing_tour(p)
{'S': 2.2666666666666666, 'N': 0.8666666666666669}
```

fundamental_matrix()

Return the fundamental matrix. [source]

See also Kemeny J. G.; Snell, J. L. Finite Markov Chains. Springer-Verlag: New York, 1976.

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.fundamental_matrix()
{('B', 'A'): 0.17751479289940991, ('A', 'B'): 0.053254437869822958,
('A', 'A'): 0.94674556213017902, ('B', 'B'): 0.82248520710059214}
```

kemeny_constant()

Return the Kemeny constant of the transition matrix. [source]

```
>>> T = pykov.Chain({('A','B'): .3, ('A','A'): .7, ('B','A'): 1.})
>>> T.Kemeny_constant()
1.7692307692307712
```

Acknowledgements

I desire to thanks my supervisor, Amedeo Caflisch, who easily understood what I needed and gave it to me. I learned a lot in the last years, enhancing my research skills and knowledge.

I also want to thanks all the stimulating people with which I shared my time. With minimalistic taste I prefer to hide the names, for sure who is in the list is aware of this.

R.